RISERVATO PERSONAL

Spedizione in Abb. Postale Gruppo III/70 - Inserto pubblicitario
Questo fascicolo è annesso alla rivista Bit n. 47 Febbraio 1984 - I.P.

APPLE

Compattazione di figure su dischetto

La grafica dell'Apple ha una risoluzione sufficientemente elevata per indurre a comporre grafici, istogrammi, prospettive, animazioni e figure di ogni genere. Il tempo necessario all'esecuzione di un programma grafico è generalmente tanto più lungo quanto maggiore è la densità e la complessità delle informazioni che la figura deve contenere; ad esempio il grafico tridimensionale di una funzione molto complessa può richiedere ore di elaborazione. Se poi esiste la possibilità che questo elaborato risulti utile anche in futuro, ci si pone il problema di memorizzarlo per non dover eseguire nuovamente l'estenuante e ormai inutile programma.

di **F. Zgavc**

Il problema di per se è facilmente risolubile, i comandi:

```
BSAVE FIGURAHGR, A$2000,L$2000
BSAVE FIGURAHGR2, A$4000,L$2000
```

permettono rispettivamente di salvare sul disco le figure contenute nella prima e nella seconda pagina grafica con i nomi "figurahgr" e "figurahgr2". Tutto va bene finchè non ci si pongono problemi di memoria, infatti se eseguiamo un CATALOG del disco su cui abbiamo salvato la figura ci accorgiamo che questa occupa ben 34 settori, un severo monito ad essere moderati con la nostra creatività!

Quando il numero delle figure da salvare è particolarmente elevato risulta quasi obbligatorio tentare di ridurre lo spazio che occupano sul disco. Le soluzioni migliori in genere presuppongono di avere a disposizione i dati finali delle varie elaborazioni che hanno portato alla figura. Dovendo memorizzare un grafico risulterà conveniente salvare i dati relativi ai singoli punti della curva, mentre in un istogramma si potranno memorizzare i valori assunti dalle ordinate nei campionamenti; naturalmente ciò comporta una notevole perdita di tempo per la stesura dei programmi sia di scrittura che di lettura, senza considerare poi che ogni caso deve essere trattato separatamente. Questo articolo propone allora come soluzione due subroutine che, pur con risultati di compattazione assai più modesti, non si vincolano in alcun modo al programma che ha costruito la figura.

Analisi del problema

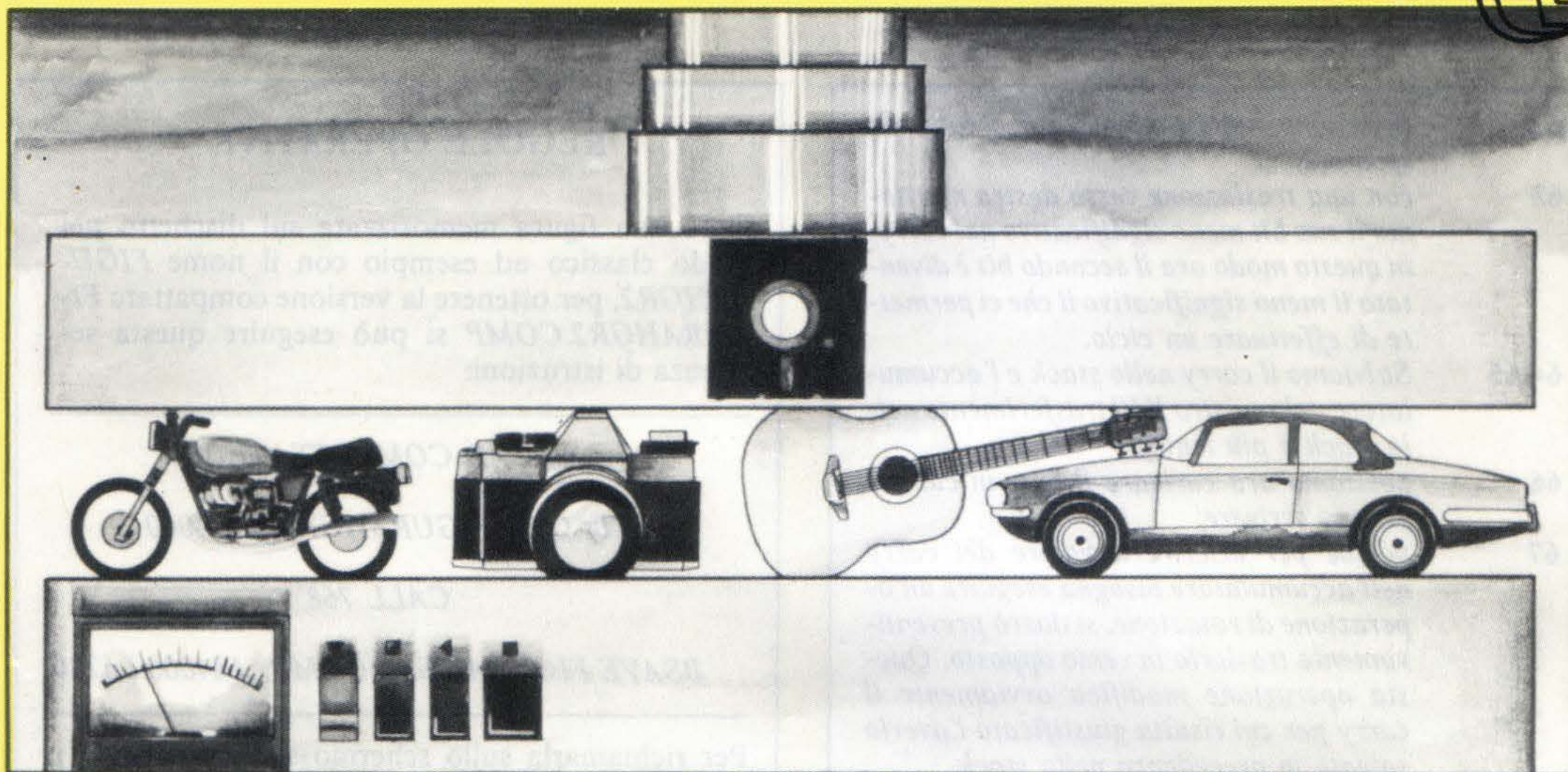
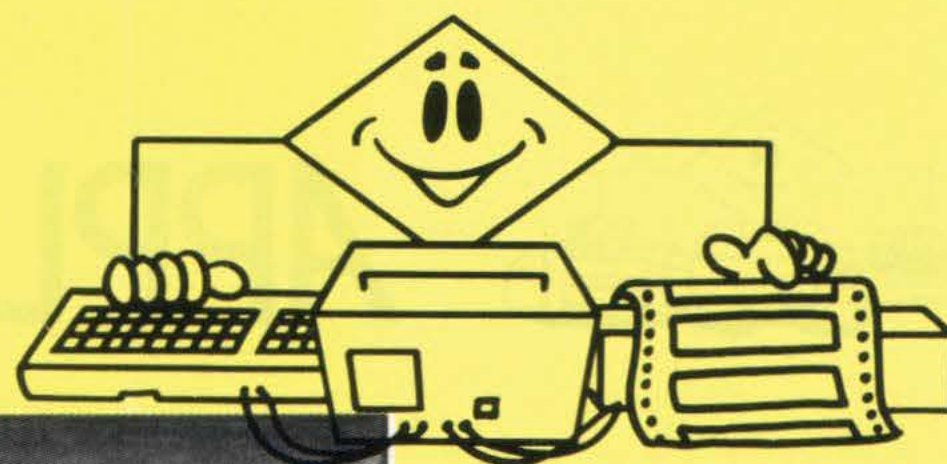
I comandi visti prima salvano sul disco una zona di memoria di lunghezza $2000 byte in esadecimale, ossia 8192 byte nella numerazione decimale. Se ricordiamo che una pagina grafica è composta da 53760 = 192x280 punti e si associa un bit ad ogni punto, otteniamo 6720 = 53760/8 byte con un disavanzo quindi di 1472 = 8192–6720 byte.

A questo punto qualcuno potrebbe obiettare che associare un bit ad ogni punto non è corretto visto che l'Apple mette a disposizione ben più di due colori per la grafica. Il nostro atteggiamento deriva essenzialmente dal metodo con cui sono effettivamente memorizzati i singoli punti ed è inoltre giustificato dalle pesanti limitazioni imposte nell'uso dei colori. Nella trattazione supporremo allora che i colori disponibili siano solo il bianco ed il nero; detto questo passiamo senz'altro a localizzare i byte intrusi per poterli poi eliminare.

Le pagine grafiche in alta risoluzione

Come risulta intuibile dai parametri indicati nel comando BSAVE la pagina HGR è posizionata nella memoria dal byte 8192 al byte 16383 e la pagina HGR2 dal byte 16384 al byte 24575. Le due pagine dunque occupano il medesimo numero di byte, infatti, contrariamente a quanto potrebbe sembrare da un'analisi superficiale, le quattro righe di testo che normalmente sono presentate nella pagina HGR non sostituiscono ma semplicemente nascondono la corrispondente porzione di grafico. Per visualizzare l'intera pagina sarà sufficiente il comando POKE-16302,0, mentre con POKE-16 301,0 si ritornerà alla situazione iniziale. Chi vuol saperne di più può consultare il "Manuale di riferimento del BASIC Applesoft" alle pagine 87-90 e 131-134 o il "manuale Apple di riferimento" alle pagine 12, 13, 19, 130. Essendo le due pagine grafiche intrinsecamente uguali a meno di una traslazione di 8192 byte nella memoria, prenderemo in esame la HGR2 che l'Applesoft offre come completamente grafica.

Per decifrare l'algoritmo che lega i punti della pagina, disposti in una struttura bidimensionale, ai byte della memoria, organizzati in modo unidimensionale, sfrutteremo il risultato grafico di alcuni micro-programmi. Cerchiamo intanto di stabilire la corrispondenza tra il contenuto di un byte e la sua rappresentazione grafica; a tal scopo eseguiamo il seguente comando:

```
HGR2:FOR I=O TO 255:POKE
16384,I:FOR J=0 TO
200:NEXT:NEXT
```

In alto a sinistra del monitor appaiono tutte le configurazioni corrispondenti ai valori da 0 a 255, ci sono sette punti, ciascuno associato ad un determinato bit del byte, più esattamente al bit meno significativo corrisponde il punto più a sinistra e gli altri seguono in ordine. Poichè l'ottavo bit non corrisponde ad alcun punto essendo riservato alla gestione del colore, si potrebbe pensare che nelle nostre ipotesi di considerare solo il bianco e il nero i valori dal 128 al 255 siano un'inutile copia dei precedenti. In realtà una differenza esiste e possiamo rendercene conto confrontando due byte con i valori sfasati di 128 unità:

```
HGR2:FOR I=0 TO 127:POKE
16384,I:POKE 17408,I
+128:FOR J=0 TO 200:
NEXT:NEXT
```

Risulta molto chiaro che il byte col valore maggiore è leggermente traslato verso destra. Questo effetto un po' particolare , che in un certo senso raddoppia la definizione orizzontale, viene spesso usato per ottenere una maggiore precisione delle lettere disegnate in pagina grafica; per tutte le altre utilizzazioni si presenta di uso molto complesso rispetto alle possibilità offerte. Allo scopo di ricuperare lo spazio corrispondente supporremo l'ottavo bit ininfluente nel grafico; anzi, per essere più precisi, daremo per scontato che esso sia nullo eliminando così eventuali effetti collaterali nella ricostruzione della figura.
Passiamo ora ad esaminare la relazione tra la posizione del byte in memoria e quella corrispondente nella pagina grafica; come prima sfrutteremo un piccolo programma (la scelta del valore 126 corrispondente ad un puntino nero e sei bianchi permette di distinguere i singoli byte):

```
HGR2:FOR I=16384 TO
24575:POKE I,126:FOR J=0
TO 50:NEXT:NEXT
```

I primi 40 byte compaiono uno di seguito all'altro nella prima riga in alto (che indichiamo con il numero 0 seguendo la numerazione propria dei comandi Applesoft) mentre il quarantunesimo si posiziona all'inizio della riga 64 seguito da altri 39 byte, l'ottantunesimo infine corrisponde all'inizio della riga 128 seguito ovviamente da altri 39. Il fatto più interessante è che i byte dal 120 al 127 non compaiono da alcuna parte e tra l'altro non sono adibiti ad alcuna funzione particolare riguardante la grafica, dunque si prestano benissimo ad essere utilizzati in altro modo.
Continuando con i byte successivi la situazione si ripete fino a riempire completamente la pagina; l'analisi dell'algoritmo di corrispondenza byte-bit - riga, colonna e viceversa non interessa la nostra trattazione per cui rimandiamo ad altri articoli o ancor meglio all'intuito del lettore.

Possiamo riassumere quanto detto con le seguenti considerazioni: nella zona di memoria corrispondente ad una pagina grafica in alta risoluzione, supponendo valide le ipotesi sopra introdotte, risultano inutilizzati 8 byte ogni 128 per un totale di 512 più un bit per ogni byte che appare sul video ossia 7680=192 x 40 bit corrispondenti a 960 byte. Abbiamo cioè localizzato quei 1472 byte di disavanzo sopra menzionati.

Compattazione delle informazioni

L'algoritmo che illustriamo, partendo da una qualsiasi figura nella seconda pagina grafica, trasla le informazioni degli ultimi byte negli spazi che abbiamo reso disponibili. Come risultato finale si ottiene una sequenza di 6720 byte che si può salvare su disco tramite il comando:

```
BSAVE FIGURAHGR2,A$4000,L6720
```

Il risparmio di memoria si concretizza così sul dischetto con la disponibilità di 6 settori dei 34 altrimenti utilizzati.

Poichè un programma in Applesoft che compatti una figura lavora imperterrito per dei minuti scoraggiando il più volonteroso utilizzatore, ricorreremo al solito Assembler tramite il quale i tempi di esecuzione si riducono a meno di un secondo. Facciamo osservare che i byte da spostare non sono necessariamente quelli da noi scelti, così come può essere modificato l'ordine con cui vengono disposti, ciò che conta invece è che il programma sia semplice, compatto e veloce.

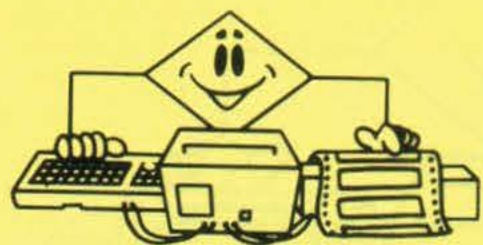



Figura 1 - Dettagli sull'Assembler del 6502.

62	*Inseriamo nell'accumulatore il byte da spostare,*
63	*con una traslazione verso destra mettiamo il suo bit meno significativo nel carry, in questo modo ora il secondo bit è diventato il meno significativo il che ci permette di effettuare un ciclo.*
64,65	*Salviamo il carry nello stack e l'accumulatore nel registro X (il trasferimento nello stack è più lento),*
66	*possiamo ora caricare il byte in cui vogliamo scrivere;*
67	*poichè per inserire il valore del carry nell'accumulatore bisogna eseguire un'operazione di rotazione, si dovrà preventivamente traslarlo in verso opposto. Questa operazione modifica ovviamente il carry per cui risulta giustificato l'averlo salvato in precedenza nello stack.*
68,69	*Recuperato il carry inseriamo il suo valore nell'accumulatore*
70	*riscriviamo il byte modificato*
71	*e portiamo nell'accumulatore il valore del registro X.*
72-74	*Possiamo senz'altro ripetere il ciclo finchè tutti e 7 i bit grafici non sono stati salvati.*

Lo spazio che dobbiamo utilizzare è diviso in due distribuzioni chiaramente diverse, quindi anche il programma di compattazione risulterà scisso in due parti. Utilizzeremo dapprima i byte che non compaiono sul video, infatti questi non pongono alcuna limitazione al tipo di grafico e conferiscono a questa prima parte la massima generalità d'uso.

Come abbiamo già accennato il nostro algoritmo sposta gli ultimi 512 byte della figura (riferendoci chiaramente all'ordine interno alla memoria); poichè questi corrispondono a 4 gruppi di 128 è evidente che ce ne sono 32 che non contengono informazioni per cui in realtà i byte da spostare sono soltanto 480 o più precisamente 4 gruppi di 120. Lo spazio in cui vogliamo sistemarli è invece costituito da 60 gruppi di 8 byte distanziati tra loro di 128 byte.

La parte che stiamo discutendo è quella intitolata "RECUPERO BYTE 120-127" del programma "COMPATTA PAGINA GRAFICA"; (listato 1) dal punto di vista delle mere istruzioni essa consiste essenzialmente di due cicli, quello esterno che incrementa di 128 unità il puntatore SCRIVI e di 8 il puntatore LEGGI e quello interno che trasla otto byte. L'unica piccola difficoltà riguarda l'incremento del puntatore LEGGI che deve evitare i byte dal 120 al 127 di ogni serie di 128. Il commento alle righe 12, 16 e 34 si riferisce ai valori da sostituire qualora si voglia compattare la prima pagina grafica anzichè la seconda.

Dopo l'esecuzione di questa prima compattazione disponiamo di 7680 byte, la seconda parte, intitolata "RECUPERO BIT 7", trasla il contenuto degli ultimi 960 nel bit più significativo dei byte precedenti. Poichè nel ciclo principale di questa seconda parte le istruzioni si differenziano alquanto dalle procedure tipiche dell'Applesoft inseriamo per chi ha già un'infarinatura sull'Assembler del 6502 una spiegazione con maggior dettaglio di informazioni. (Figura 1)

REGOLE OPERATIVE

Data una figura memorizzata sul dischetto nel modo classico ad esempio con il nome *FIGURAHGR2*, per ottenere la versione compattata *FIGURAHGR2.COMP* si può eseguire questa sequenza di istruzioni:

BLOAD COMPATTA.OBJ0

BLOAD FIGURAHGR2, A$ 4000

CALL 768

BSAVE FIGURAHGR2.COMP, A$4000,L6720

Per richiamarla sullo schermo sono sufficienti i comandi:

BLOAD ESPANDI.OBJ0

HGR2:PRINT CHR$ (4)
"BLOAD FIGURAHGR2.COMP"
:CALL 768

La rimanente parte del programma non pone difficoltà alcuna di interpretazione, ci limiteremo a ricordare che nelle righe 54, 58 e 87 sono indicati i valori da sostituire se si vuole lavorare nella prima pagina grafica.

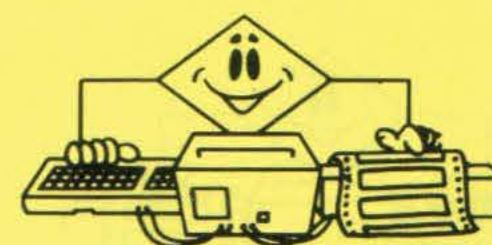
Espansione delle informazioni

La figura compattata di cui ora disponiamo può essere riletta con il solito comando:

```
BLOAD FIGURAHGR2
```

Naturalmente è necessario espandere le informazioni contenute ripercorrendo all'indietro la compattazione precedentemente eseguita: "ESPANDI PAGINA GRAFICA" (listato 2) assolve appunto a questo compito. Vista la costruzione sostanzialmente simile al programma precedente, ci limitiamo a far notare che i bit più significativi di ogni byte vengono azzerati in conformità a quanto stabilito nell'introduzione.

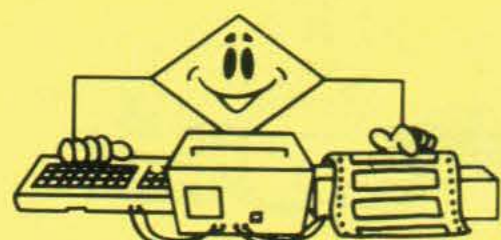
Prima di concludere commentiamo sommariamente il funzionamento dei programmi e i risultati ottenuti. Poichè non sono stati utilizzati dei salti incondizionati i programmi sono perfettamente rilocabili, ossia possono essere caricati ovunque nella memoria. La lunghezza molto contenuta dei listati, per quanto a discapito della chiarezza si potrebbe rosicchiare ancora qualche byte, permette di caricare le subroutine alla locazione $300 ossia 768 in decimale; volendo disporre di entrambe le subroutine contemporaneamente si potrà caricarne una o entrambe alla locazione $6000 (24576 in decimale). Naturalmente se nel programma Applesoft è presente un'istruzione del tipo:

```
LOMEM:24576
```

andrà modificata in modo da tener conto dello spazio occupato dalle subroutine inserite.

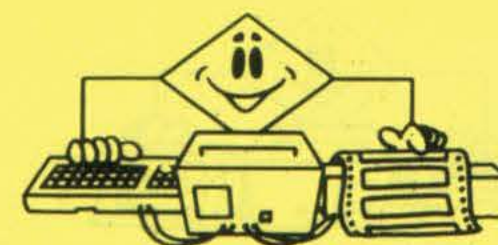
Al solito, per chi non dispone di un assemblatore, vengono riportate anche le sequenze di codici da inserire direttamente tramite il monitor dell'Apple per il controllo potrà risultare utile confrontare il programma disassemblato con i listati dell'articolo. Un'ultima nota riguarda il richiamo delle subroutine che si effettua tramite un CALL al primo byte delle stesse.

Lo spazio risparmiato con questa compattazione all'incirca al 18% del totale, il che è abbastanza se si considera che i tempi necessari per salvare o rileggere una figura sono identici se non inferiori a quelli del metodo classico. Ci sono naturalmente altre vie per risparmiare spazio, ad esempio quando si usa la prima pagina con le righe di testo la parte nascosta è ovviamente del tutto inutile, chissà che in un prossimo articolo...

Listato 1 - Programma di compattazione.

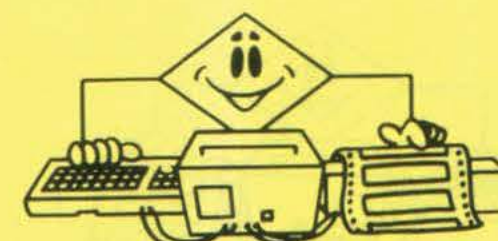
```
 1 ;=================================
 2 ;COMPATTA              PAGINA GRAFICA
 3 ;=================================
 4 LEGGI     EQU  $FC
 5 SCRIVI    EQU  $FE
 6           ORG  $300
 7 ;=================================
 8 ;RECUPERO               BYTE 120-127
 9 ;=================================
10           LDY  #120      SCRIVI=16384+120
11           STY  SCRIVI
12           LDY  #$40      (HGR #$20)
13           STY  SCRIVI+1
14           LDY  #0        LEGGI=24576-4*128
15           STY  LEGGI
16           LDY  #$5E      (HGR #$3E)
17           STY  LEGGI+1
18 ;=================================
19 INIZIO1   LDY  #0
20 CICLO1    LDA  (LEGGI),Y
21           STA  (SCRIVI),Y
22           INY
23           CPY  #8
24           BNE  CICLO1
25 ;=================================
26           CLC            LEGGI=LEGGI+8
27           LDA  LEGGI
28           ADC  #8
29           CMP  #120      FINE RIGA?
30           BEQ  CONT1
31           CMP  #248
32           BNE  CONT2
33           LDA  LEGGI+1   FINE?
34           CMP  #$5F      (HGR #$3F)
35           BEQ  CONT3
36           INC  LEGGI+1
37           LDA  #0
38           BEQ  CONT2
39 CONT1     LDA  #128
40 CONT2     STA  LEGGI
41 ;=================================
42           CLC            SCRIVI=SCRIVI+128
```

APPLE

Seguito listato 1.

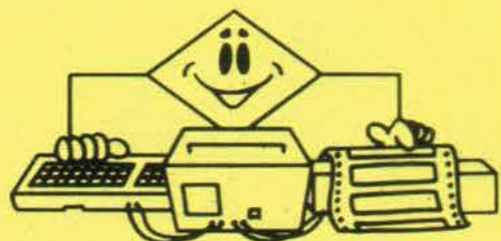
```
43            LDA   #128
44            ADC   SCRIVI
45            STA   SCRIVI
46            BCC   INIZIO1
47            INC   SCRIVI+1
48            BCS   INIZIO1
49 ;=================================
50 ;RECUPERO                  BIT 7
51 ;=================================
52 CONT3      LDY   #$40       LEGGI=16384+6720
53            STY   LEGGI
54            LDY   #$5A       (HGR #$3A)
55            STY   LEGGI+1
56            LDY   #$0        SCRIVI=16384
57            STY   SCRIVI
58            LDY   #$40       (HGR #$20)
59            STY   SCRIVI+1
60 ;=================================
61 INIZIO2    LDY   #0
62            LDA   (LEGGI),Y
63 CICLO2     LSR   A          BIT 0 NEL CARRY
64            TAX              SALVA A
65            PHP              SALVA CARRY
66            LDA   (SCRIVI),Y
67            ASL   A
68            PLP
69            ROR   A          BIT 7=CARRY
70            STA   (SCRIVI),Y
71            TXA
72            INY
73            CPY   #7
74            BNE   CICLO2
75 ;=============================
76            INC   LEGGI      LEGGI=LEGGI+1
77            BNE   CONT4
78            INC   LEGGI+1
79 CONT4      CLC              SCRIVI=SCRIVI+7
80            LDA   #7
81            ADC   SCRIVI
82            STA   SCRIVI
83            BCC   CONT5
84            INC   SCRIVI+1
85 CONT5      CMP   #$40       FINE?
86            BNE   INIZIO2
87            LDA   #$5A       (HGR #$3A)
88            CMP   SCRIVI+1
89            BNE   INIZIO2
90            RTS
```

Listato 2 - Programma di espansione.

```
 1 ;==================================
 2 ;ESPANDI PAGINA GRAFICA
 3 ;==================================
 4 LEGGI     EQU  $FC
 5 SCRIVI    EQU  $FE
 6           ORG  $300
 7 ;==================================
 8 ;ESPANDI BIT7
 9 ;==================================
10           LDY  #0           LEGGI=16384
11           STY  LEGGI
12           LDY  #$40         (HGR #$20)
13           STY  LEGGI+1
14           LDY  #$40         SCRIVI=16384+6720
15           STY  SCRIVI
16           LDY  #$5A         (HGR #$3A)
17           STY  SCRIVI+1
18 ;==================================
19 INIZIO1   LDX  #0
20           LDY  #6
21 CICLO1    LDA  (LEGGI),Y
22           ASL  A            BIT 7 NEL CARRY
23           PHA               SALVA A
24           TXA
25           ROL  A            BIT 0=CARRY
26           TAX
27           PLA
28           LSR  A            BIT 7=0
29           STA  (LEGGI),Y
30           DEY
31           BPL  CICLO1
32           BEQ  CICLO1
33 ;==================================
34           TXA
35           LDY  #0
36           STA  (SCRIVI),Y
37           INC  SCRIVI       SCRIVI=SCRIVI+1
38           BNE  CONT1
39           INC  SCRIVI+1
40 CONT1     LDA  LEGGI        LEGGI=LEGGI+7
41           CLC
42           ADC  #7
43           STA  LEGGI
44           BCC  CONT2
45           INC  LEGGI+1
46 CONT2     CMP  #$40
47           BNE  INIZIO1      FINE?
```

Seguito listato 2.

```
48              LDA   #$5A
49              CMP   LEGGI+1
50              BNE   INIZIO1
51 ;================================
52 ;ESPANDI BYTE 120-127
53 ;================================
54              LDY   #0          SCRIVI=24576-4*128
55              STY   SCRIVI
56              LDY   #$5E        (HGR #$3E)
57              STY   SCRIVI+1
58              LDY   #120        LEGGI=16384+120
59              STY   LEGGI
60              LDY   #$40        (HGR #$20)
61              STY   LEGGI+1
62 ;================================
63 INIZIO2      LDY   #0
64 CICLO2       LDA   (LEGGI),Y
65              STA   (SCRIVI),Y
66              INY
67              CPY   #8
68              BNE   CICLO2
69 ;================================
70              CLC               SCRIVI=SCRIVI+8
71              LDA   SCRIVI
72              ADC   #8
73              CMP   #120        FINE RIGA?
74              BEQ   CONT4
75              CMP   #248
76              BNE   CONT5
77              LDA   SCRIVI+1    FINE?
78              CMP   #$5F        (HGR #$3F)
79              BNE   CONT3
80              RTS
81 CONT3        INC   SCRIVI+1
82              LDA   #0
83              BEQ   CONT5
84 CONT4        LDA   #128
85 CONT5        STA   SCRIVI
86 ;================================
87              CLC               LEGGI=LEGGI+128
88              LDA   #128
89              ADC   LEGGI
90              STA   LEGGI
91              BCC   INIZIO2
92              INC   LEGGI+1
93              BCS   INIZIO2
```

```
0300- A0 78 84 FE A0 40 84 FF
0308- A0 00 84 FC A0 5E 84 FD
0310- A0 00 B1 FC 91 FE C8 C0
0318- 08 D0 F7 18 A5 FC 69 08
0320- C9 78 F0 10 C9 F8 D0 0E
0328- A5 FD C9 5F F0 17 E6 FD
0330- A9 00 F0 02 A9 80 85 FC
0338- 18 A9 80 65 FE 85 FE 90
0340- CF E6 FF B0 CB A0 40 84
0348- FC A0 5A 84 FD A0 00 84
0350- FE A0 40 84 FF A0 00 B1
0358- FC 4A AA 08 B1 FE 0A 28
0360- 6A 91 FE 8A C8 C0 07 D0
0368- F0 E6 FC D0 02 E6 FD 18
0370- A9 07 65 FE 85 FE 90 02
0378- E6 FF C9 40 D0 D7 A9 5A
0380- C5 FF D0 D1 60
```

```
0300- A0 00 84 FC A0 40 84 FD
0308- A0 40 84 FE A0 5A 84 FF
0310- A2 00 A0 06 B1 FC 0A 48
0318- 8A 2A AA 68 4A 91 FC 88
0320- 10 F2 F0 F0 8A A0 00 91
0328- FE E6 FE D0 02 E6 FF A5
0330- FC 18 69 07 85 FC 90 02
0338- E6 FD C9 40 D0 D2 A9 5A
0340- C5 FD D0 CC A0 00 84 FE
0348- A0 5E 84 FF A0 78 84 FC
0350- A0 40 84 FD A0 00 B1 FC
0358- 91 FE C8 C0 08 D0 F7 18
0360- A5 FE 69 08 C9 78 F0 11
0368- C9 F8 D0 0F A5 FF C9 5F
0370- D0 01 60 E6 FF A9 00 F0
0378- 02 A9 80 85 FE 18 A9 80
0380- 65 FC 85 FC 90 CE E6 FD
0388- B0 CA
```

Listato 3 - Codici esadecimali relativi ai due programmi. A sinistra il programma di compattazione, a destra quello di espansione.

Applesoft strutturato: arricchiamo il BASIC Applesoft

Di programmazione strutturata, e dei benefici non solo dal punto di vista pratico, ma anche da quello educativo, che molti sostengono possa procurare a chi ne fa uso, si è parlato e scritto parecchio.
Chi però ha imparato a programmare direttamente in BASIC sul proprio personal, avrà probabilmente tuttora idee molto vaghe in proposito, come sempre avviene quando di qualcosa si è sentito parlare ma non la si è sperimentata direttamente.

di **G. Azzali**

Bene, ecco un programma che permette, a chiunque possieda un Apple II dotato del solo Applesoft, di "arricchire" il BASIC di questo personal con alcune istruzioni tipiche della programmazione strutturata.
Per poterlo utilizzare è necessario disporre anche della routine PGM1 <—> PGM2, presentata su questa rivista nella rubrica "Il ricettario - Apple a mezzadria".

Il programma funziona come "traduttore": esamina il programma di partenza, che può essere costituito da tutte le normali istruzioni Applesoft e in più dalle istruzioni "aggiunte" di tipo strutturato, e lo trasforma in un normale programma BASIC. Quindi lo scopo di questo traduttore non è quello di "emulare" linguaggi più evoluti, ma, più semplicemente, di facilitare la programmazione in BASIC. Per chi già conosce l'Applesoft, occorre solo imparare qualche nuova istruzione; inoltre, il prodotto della traduzione potrà "girare" su qualunque Apple senza più bisogno del traduttore.

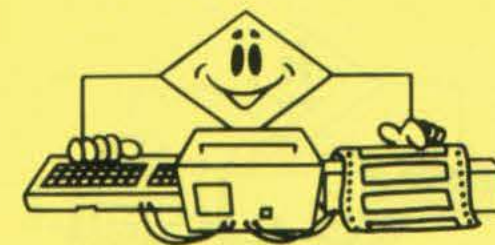
Le istruzioni "supplementari" sono le seguenti: REPEAT-UNTIL, WHILE-WEND, IF-THEN-(ELSE)-ENDIF. Prima di esaminarne il significato, precisiamo alcune importanti limitazioni:
— tutte queste parole chiave devono essere racchiuse tra doppi apici (in caso contrario, infatti, il BASIC tenterebbe di interpretarle secondo le proprie regole: REPEAT, ad esempio, verrebbe spezzato nella variabile REPE e nella parola chiave AT),
— non devono essere lasciati spazi nè tra le singole lettere di una parola chiave nè tra gli apici e la parola chiave stessa,
— le parole chiave "aggiunte" devono occupare da sole una linea di programma (non è permesso, cioè, concatenarle ad altre istruzioni mediante i due punti).
Vediamo ora la sintassi e il significato delle varie strutture, nonchè la loro traduzione in BASIC:
1) struttura REPEAT-UNTIL

```
10    "REPEAT"
20                S1
30                S2
 .                 .
 .                 .
 .                 .
100               Sn
110   "UNTIL" expr
```

Significato: la sequenza di istruzioni S1...Sn viene ripetuta finchè l'espressione logica "expr" non diventa vera; il test su "expr" viene eseguito dopo la sequenza S1 ...Sn.
(Nota: gli statement S1 ... Sn sono stati scritti incolonnati a destra del REPEAT per evidenziare la struttura, ma in pratica si potranno scrivere normalmente all'inizio della riga, o anche più di uno sulla stessa riga separati da due punti; questo varrà anche per i prossimi esempi).
La struttura REPEAT-UNTIL viene tradotta così:

```
10 REM REPEAT
20 S1
30 S2
 .
 .
100 Sn
110 IF NOT (expr) THEN 10
```

2) struttura WHILE-WEND

```
10    "WHILE" expr
20                S1
30                S2
 .                 .
 .                 .
 .                 .
100               Sn
110     "WEND"
```

Significato: la sequenza di istruzioni S1 ... Sn viene ripetuta finchè l'espressione logica "expr" si mantiene vera; il test su "expr" viene eseguito prima della sequenza S1 ... Sn.

La struttura WHILE-WEND viene tradotta così (con XXX si è indicato il numero di linea dell'istruzione successiva a WEND):

```
10 IF NOT (expr)
   THEN XXX
20 S1
30 S2
.
.
.
100 Sn
110 GOTO 10
XXX
```

Nel caso non vi sia un'istruzione successiva a WEND (cioè WEND è l'ultima istruzione di un programma), al posto del numero di linea XXX viene messo un END.

3) struttura IF-THEN-(ELSE)-ENDIF

Ci si potrebbe chiedere il motivo per cui non è stata inserita la sola parola chiave ELSE, dal momento che il BASIC possiede già l'istruzione IF-THEN. La spiegazione è che l'IF strutturato consente di superare la limitazione del BASIC di dover scrivere tutte le istruzioni conseguenti all'IF nella stessa linea di programma. In BASIC, infatti, si deve scrivere:
IF expr THEN S1:S2:...:Sn,
dove S1...Sn sono le istruzioni da eseguire se "expr" è vera. Questo modo di scrivere può non essere attuabile se la sequenza S1 ... Sn è troppo lunga, o se uno degli statement S1 ... Sn è a sua volta un IF-THEN, e bisogna allora ricorrere a scomode "perifrasi".

La forma strutturata dell'istruzione IF, invece, consente di scrivere un numero illimitato di linee di programma da eseguire se la condizione è vera, e un numero illimitato di linee di programma da eseguire se la condizione è falsa (ELSE). Naturalmente, rimane ancora utilizzabile la forma normale IF...THEN, che si distingue dalla precedente per la mancanza dei doppi apici.
Ecco come va usato l'IF strutturato:

```
10    "IF" expr "THEN"
20              S1
.               .
.               .
.               .
100             Sn
110       "ELSE"
120             S'1
.               .
.               .
.               .
200             S'n
210     "ENDIF"
```

Significato: se l'espressione logica "expr" è vera, vengono eseguite le istruzioni S1 ... Sn, altrimenti le istruzioni S'1 ... S'n.
(Se non vi sono le istruzioni S'1 ... S'n, ELSE può essere omesso)
La struttura IF-THEN-ELSE viene tradotta così:

```
10 IF NOT (expr) THEN 120
20 S1
.
.
.
100 Sn
110 GOTO 210
120 S'1
.
.
.
200 S'n
210 REM ENDIF
```

Nelle figure 1 e 2 sono riportati due semplici programmi, che fanno uso di tutte le strutture descritte, prima e dopo la traduzione (il primo calcola il M.C.D. tra due numeri con il metodo di Euclide, il secondo stampa la ... tavola pitagorica).

```
10  REM PROGRAMMA EUCLIDE
20  REM CALCOLA IL M.C.D.
30  REM TRA DUE NUMERI
40  REM
50  INPUT "A,B? ";I,J
60 "WHILE"I <  > J
70 "IF"I > J"THEN"
80 I = I - J
90 "ELSE"
100 J = J - I
110 "ENDIF"
120 "WEND"
130  PRINT "M.C.D.=";I
```

Figura 1 - Programma Euclide *nella versione Applesoft esteso.*

```
10  REM PROGRAMMA EUCLIDE
20  REM CALCOLA IL M.C.D.
30  REM TRA DUE NUMERI
40  REM
50  INPUT "A,B? ";I,J
60  IF  NOT (I <  > J) THEN 130
70  IF  NOT (I > J) THEN 100
80 I = I - J
90  GOTO 110
100 J = J - I
110  REM ENDIF
120  GOTO 60
130  PRINT "M.C.D.=";I
```

Figura 1a - Programma Euclide *dopo la traduzione (tempo di traduzione circa 6 secondi).*

Figura 2 - Programma tavola Pitagorica *nella versione Applesoft esteso.*

```
10000  REM TAVOLA PITAGORICA
10005  REM
10010  HOME
10020 A = 1
10030 "REPEAT"
10040 B = 1
10050 "REPEAT"
10060  PRINT  TAB( 3 * B)B * A;
10070 B = B + 1
10080 "UNTIL"B = 10
10085  PRINT : PRINT
10090 A = A + 1
10100 "UNTIL"A = 10
10110  PRINT : PRINT
```

Figura 2a - Programma tavola Pitagorica *dopo la traduzione (tempo di traduzione 5 secondi).*

```
10000  REM TAVOLA PITAGORICA
10005  REM
10010  HOME
10020 A = 1
10030  REM REPEAT
10040 B = 1
10050  REM REPEAT
10060  PRINT  TAB( 3 * B)B * A;
10070 B = B + 1
10080  IF NOT (B = 10) THEN 1005
     0
10085  PRINT : PRINT
10090 A = A + 1
10100 IF  NOT (A = 10) THEN 1003
     0
10110  PRINT : PRINT
```

Descrizione del programma

Come anticipato, il programma "traduttore" richiede che sia stata preliminarmente caricata la routine PGM1<—>PGM2, che consente di allocare due programmi BASIC in memoria. Questo si rende necessario per il fatto che sia il "traduttore", sia il programma da tradurre, sono scritti in BASIC (il secondo è scritto in "BASIC esteso", ma ciò non fa differenza ai fini del caricamento in memoria). Per quanto riguarda l'uso della routine PGM1 <—> PGM2 si rimanda all'articolo in cui la routine stessa è stata descritta; qui si precisa soltanto che il "programma 2" (allocato in posizione più alta) sarà costituito dal traduttore, mentre il "programma 1" sarà quello da tradurre. Per avviare la traduzione si deve richiamare il programma 2 con CALL768 e dare il RUN.

In figura 3 è riportato il listato del "traduttore BASIC esteso → BASIC". Il programma principale è costituito dalle righe 300-350, e svolge la funzione di esaminare una ad una le linee del programma da tradurre. Quando viene incontrata una parola chiave tra doppi apici, si chiama una routine (linee 1000-1210) che la confronta con le parole chiave del "BASIC esteso" memorizzate in una tabella; se viene riconosciuta una parola valida, l'esecuzione passa alla parte di programma preposta alla traduzione di quell'istruzione.

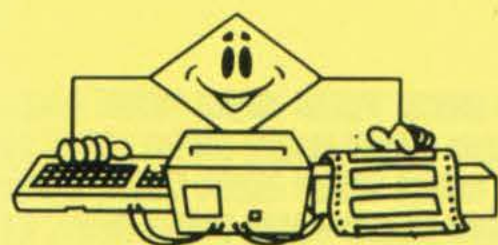
La traduzione vera e propria si ottiene modificando con dei POKE le istruzioni del programma di partenza; se la traduzione è più lunga dell'originale, bisogna anche rilocare la parte di programma che va da quel punto alla fine. Questa operazione (compiuta dalla routine CREASPAZI, linee 10000-10200) è particolarmente gravosa in quanto, oltre allo spostamento del programma, è anche necessario riaggiornare per ogni istruzione il puntatore alla successiva, nonchè modificare i puntatori di fine programma, inizio variabili semplici ecc., che la routine PGM1<—>PGM2 "salva" nella zona di memoria da $352 a $361.

Se invece l'istruzione tradotta è più breve di quella originale, anzichè rilocare il programma (operazione che comporta un notevole rallentamento dell'esecuzione) si preferisce "riempire i buchi" dell'istruzione precedente con dei codici privi di effetto (ad esempio REPEAT viene trasformato in REM REPEAT cambiando il codice degli apici di sinistra con il codice del REM e il codice degli apici di destra con quello del blank).

Per ciascun tipo di struttura è necessario l'uso di uno stack, per memorizzare il numero di linea a cui si dovrà tornare quando si incontra l'istruzione di "chiusura" della struttura (ad esempio, quando si incontra 100 REPEAT il numero di linea 100 viene inserito nello stack, per essere poi utilizzato in corrispondenza dell'UNTIL). Nel caso del WHILE ciò non è possibile, perchè quando lo si incontra bisognerebbe già conoscere il numero di linea del WEND, che però viene dopo, e sarebbe quindi necessaria una traduzione "in due passate". L'ostacolo viene superato inserendo nello stack, anzichè il numero di linea, la locazione di memoria in cui si dovrà "pokare" tale numero di linea quando lo si sarà incontrato. Il numero massimo di "livelli di annidamento" delle strutture è 20, ma può essere aumentato modificando semplicemente le istruzioni di DIMensionamento alle linee 100 e 110.

È previsto un certo numero di messaggi d'errore che segnalano, limitatamente alle istruzioni "aggiunte", sia errori di sintassi, sia errori di struttura (ad esempio un UNTIL incontrato prima di un REPEAT). Un errore rilevato a traduzione avanzata, però, porta ad un programma tradotto solo a metà che può dare degli inconvenienti quando si tenta di riavviare la traduzione; è consigliabile perciò salvare il programma fin dall'inizio (cioè in versione "Applesoft esteso"), in modo da poter ricominciare da capo se necessario.

A traduzione terminata (o in caso di errore) il programma effettua una CALL768 e perciò il passaggio programma 2 → programma 1 avviene automaticamente (dando il comando LIST si vedrà direttamente il programma tradotto).

Chiudiamo per ora il discorso sulle applicazioni della tecnica dei "due programmi in memoria", anticipando però che sono in fase di realizzazione anche un "editor di linea" e un programma per la stampa "ordinata" dei listati, entrambi basati su questa tecnica.

APPLE

LISTA DELLE VARIABILI UTILIZZATE

MAX	Numero massimo di parole chiave riconoscibili.
NL	Numero di linea dell'istruzione corrente.
CO	Codice operativo dell'istruzione esaminata.
I,J	Indici generici dei cicli.
II	Indirizzo iniziale dell'istruzione corrente.
PI	Indirizzo istruzione successiva.
N1	Puntatore generico all'interno dell'istruzione.
N2	Puntatore all'ultimo carattere di una parola chiave aggiunta.
N3	Puntatore al primo indirizzo da incrementare in CREASPAZI.
N4	Puntatore al numero di linea che CONVERTINUMERO deve trattare.
LN	Variabile locale di CONVERTINUMERO contenente il numero di linea.
UB	Indirizzo ultimo byte di memoria occupato dal programma.
BS	Indirizzo a partire dal quale CREASPAZI deve rilocare il programma.
BN	Numero di byte che CREASPAZI deve liberare.
T,T1,T2	Variabili temporanee generiche.
DV	Fattore di divisione in CONVERTINUMERO.
SG	Flag indicante cifra significativa.
ER	Variabile logica: è TRUE se si riscontra un errore nella sintassi di THEN.
CV	Numero di cifre valide restituite da CONVERTINUMERO.
KEY%(6,6)	Contiene i codici ASCII delle parole chiave e il loro numero di caratteri.
NUM%(4)	Contiene le 5 cifre di un numero di linea (codificate ASCII); se il numero di linea ha meno di 5 cifre, le cifre mancanti sono dei blank.
TH%(5)	Contiene i codici ASCII che compongono la parola chiave THEN.
RS%(20,4)	Stack contenente i numeri di linea delle istruzioni REPEAT.
C1%(20)	N. di cifre valide dei numeri di linea contenuti in RS%(20,4).
WS(20)	Stack contenente le locazioni di memoria associate ai WHILE.
WS%(20,4)	Stack contenente i numeri di linea associati ai WHILE.
IW(20)	Stack contenente le locazioni di memoria iniziali delle istruzioni WHILE.
IS(20)	Stack contenente le locazioni di memoria associate alle istruzioni IF.
RS,WS,IS	Stack pointer.

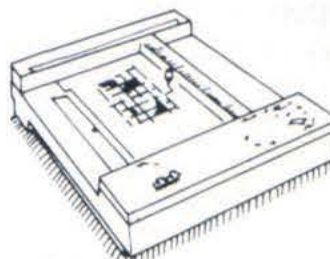



Figura 3.

COME È ... STRUTTURATO L'APPLESOFT
(dedicato agli applisti neofiti)

Se, dopo aver caricato un programma BASIC, potessimo "vedere" come si presenta la relativa area di memoria (cioè quella che normalmente va dalla locazione decimale 2048 in poi) troveremmo una disposizione simile a quella in figura.
La locazione iniziale 2048, pur non facendo parte del programma vero e proprio, deve necessariamente contenere il valore zero; con la 2049 hanno inizio le istruzioni, che sono costituite ciascuna da 4 byte iniziali, da una parte centrale variabile e da un byte di chiusura.
I primi 2 byte contengono un puntatore all'inizio dell'istruzione successiva, gli altri due contengono il "numero di linea". Leggendo il contenuto dei primi due, il BASIC è in grado di passare da un'istruzione all'altra senza doverle esaminare per intero (questo fatto è sfruttato anche dal programma "traduttore").
La parte centrale contiene invece l'istruzione vera e propria (che può in realtà essere formata da più istruzioni, visto che il BASIC consente di metterne più d'una nella stessa linea di programma). Bisogna osservare che, quando si preme *RETURN* dopo aver battuto una riga di programma, essa non viene ricopiata in memoria così com'è scritta, cioè come insieme di caratteri ASCII, ma viene prima sottoposta ad un lavoro di "compattazione": gli spazi superflui vengono eliminati, e le "parole chiave" del BASIC vengono rappresentate mediante un codice di un solo byte (cosicchè *RESTORE* e *IF*, ad esempio, occupano lo stesso spazio in memoria). La tabella completa di questi codici ("token") si trova a pag. 121 del manuale Applesoft. I valori di questi "token" cadono al di fuori del campo dei codici ASCII; in questo modo una parola chiave è immediatamente distinguibile da un nome di variabile, da un carattere di separazione, o da un numero (i numeri vengono ricopiati in memoria cifra per cifra: ogni cifra è codificata ASCII ed occupa un byte). La "compattazione" non viene invece eseguita su quanto è racchiuso tra doppi apici o su ciò che segue un *REM* o un *DATA*.
Nell'ultimo byte di ogni istruzione troviamo uno zero che serve come "marca" di fine linea. Il termine dell'intero programma è a sua volta indicato dalla presenza di due zeri nei byte che seguono l'ultima istruzione, dove dovrebbe trovarsi il puntatore alla linea di programma successiva (che in questo caso non esiste).
Sia l'indirizzo d'inizio sia quello finale del programma sono contenuti in appositi puntatori di pagina zero: il primo nelle locazioni \$67 e \$68, il secondo in \$DAF e \$B0.
Subito dopo la fine del programma comincia lo spazio riservato alle variabili (la cui "mappa", cioè lo schema della rappresentazione in memoria, è riportata a pag. 137 del manuale Applesoft). Anche per la locazione iniziale delle variabili esiste un puntatore (*LOMEM*), situato in \$69 e \$6A, che normalmente coincide con il termine del programma, ma che può essere modificato con l'istruzione *LOMEM*: qualora sia necessario proteggere l'area delle variabili da possibili "sporcamenti" (come può avvenire in caso di sovrapposizione con una pagine grafica ad alta risoluzione).

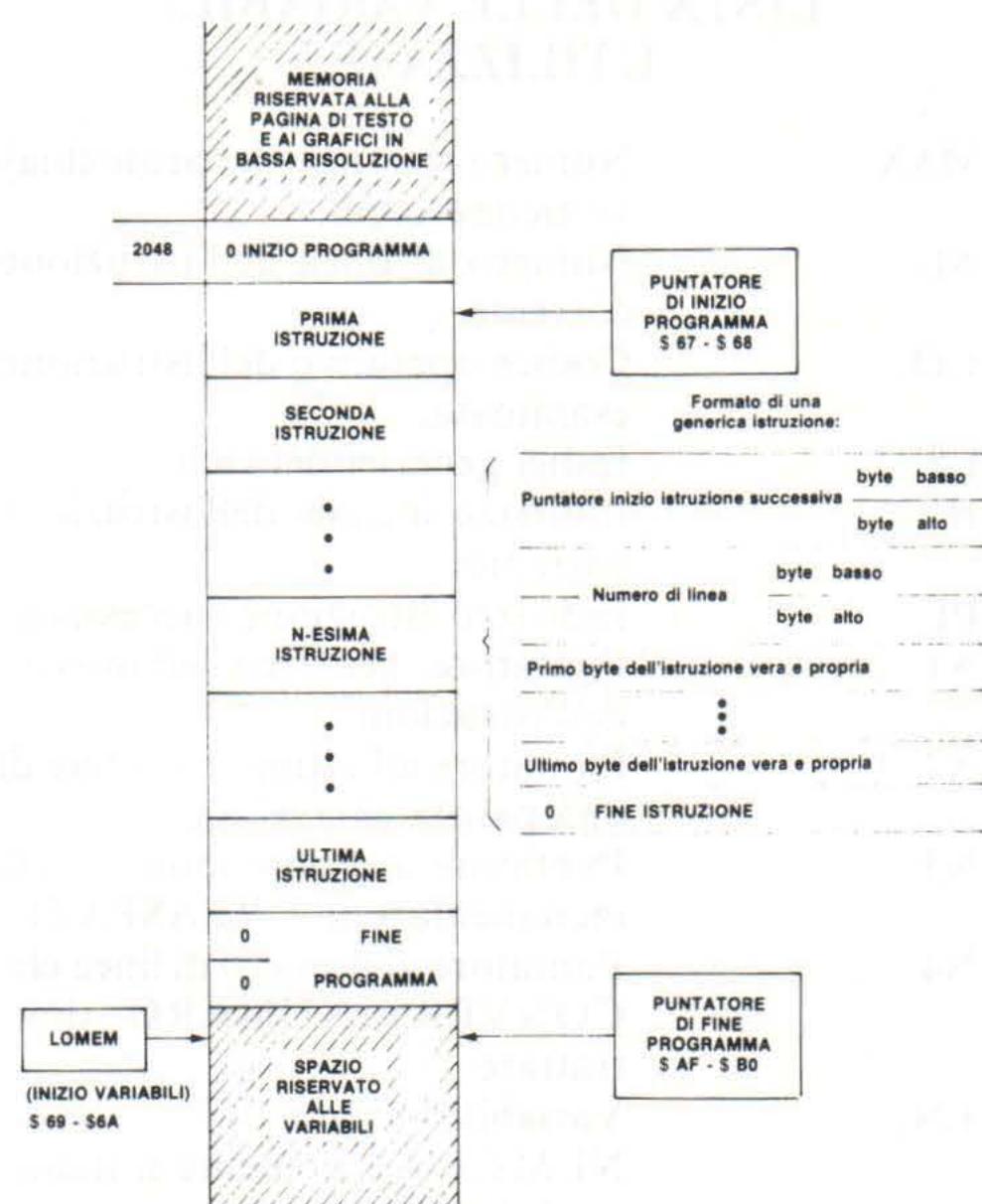
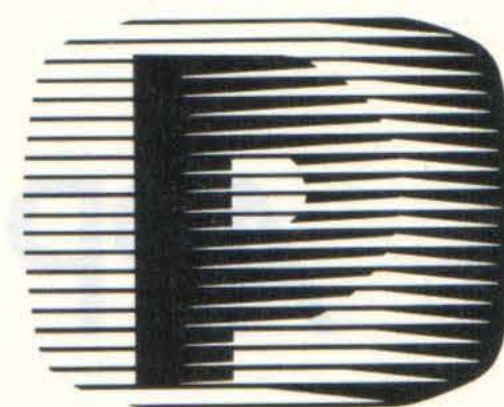
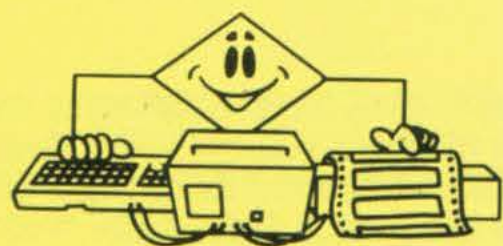
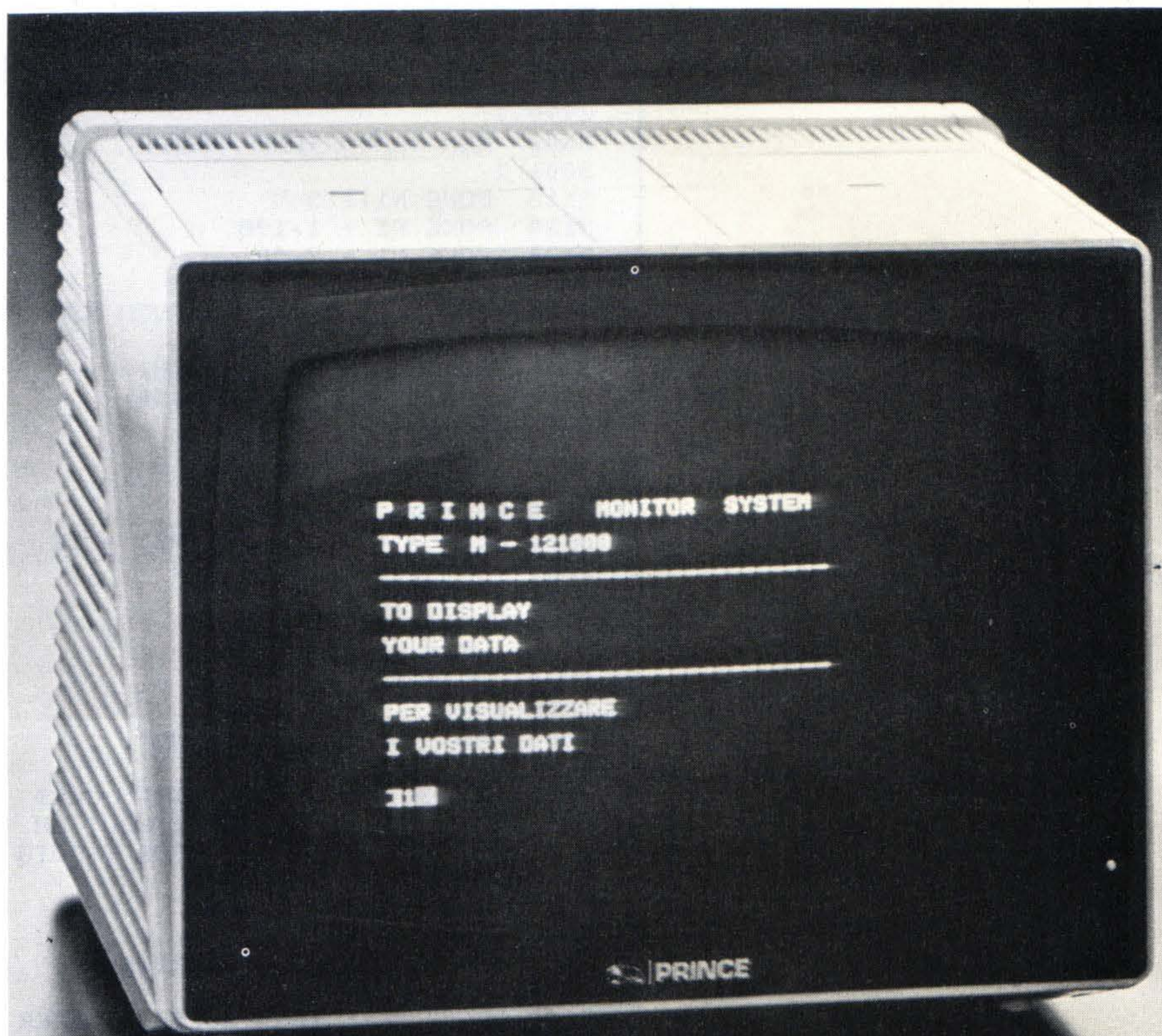



Figura 4 - Il listato BASIC.

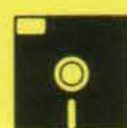
```
10  REM %%%%%%%%%%%%%%%%%%%%%%%%%
20  REM %                       %
30  REM % T R A D U T T O R E   %
35  REM %                       %
40  REM % BASIC ESTESO ->BASIC  %
45  REM %                       %
50  REM %      PER APPLE II     %
55  REM %                       %
60  REM %   G. AZZALI - 1982    %
70  REM %                       %
80  REM %%%%%%%%%%%%%%%%%%%%%%%%%
85  REM
90  TEXT
100  DIM NUM%(4),RS%(20,4),C1%(20
   ),KEY%(6,6),TH%(5)
110  DIM WS(20),WS%(20,4),IW(20),
   IS(20)
120  FOR I = 0 TO 6
130  FOR J = 0 TO 6
140  IF J > KEY%(I,0) THEN 160
150  READ KEY%(I,J)
160  NEXT : NEXT
170  FOR I = 0 TO 5: READ TH%(I):
   NEXT
180 UB =  PEEK (864) + 256 *  PEEK
   (865)
199 :
200  REM PAROLE CHIAVE
201 :
205  REM REPEAT
210  DATA 6,82,69,80,69,65,84
215  REM UNTIL
220  DATA 5,85,78,84,73,76
225  REM WHILE
230  DATA 5,87,72,73,76,69
235  REM WEND
240  DATA 4,87,69,78,68
```

Seguito figura 4.

```
245  REM IF
250  DATA 2,73,70
255  REM ELSE
260  DATA 4,69,76,83,69
265  REM ENDIF
270  DATA 5,69,78,68,73,70
275  REM THEN
280  DATA 34,84,72,69,78,34
289 :
290  REM ESAME PROGRAMMA
291 :
300 II = 2049
310 PI =  PEEK (II) +  PEEK (II +
     1) * 256
315  IF PI = 0 THEN  PRINT : PRINT
     "TRADUZIONE COMPLETATA": GOTO
     60000
320 CO =  PEEK (II + 4)
330  IF CO = 34 THEN  GOSUB 1000
350 II = PI: GOTO 310
999 :
1000  REM CONFRONTO PAROLE CHIAVE

1001 :
1050 NL =  PEEK (II + 2) +  PEEK
     (II + 3) * 256
1100 N1 = II + 4
1110 MAX = 6
1115 I = 0:J = 1
1120  IF  PEEK (N1 + J) <  > KEY%
     (I,J) THEN 1150
1130  IF J >  = KEY%(I,0) AND  PEEK
     (N1 + J + 1) = 34 THEN 1200
1135  IF J = KEY%(I,0) THEN 1150
1140 J = J + 1: GOTO 1120
1150 I = I + 1:J = 1
1160  IF I > MAX THEN  PRINT  CHR$
     (7): PRINT "?SYNTAX ERROR IN
     ";NL: POP : GOTO 60000
1170  GOTO 1120
1200 N2 = N1 + J + 1
1210  ON I GOTO 2000,3000,4000,50
     00,6000,7000
1299 :
1300  REM TRADUZIONE "REPEAT"
1301 :
1310  POKE N1,178
1320  POKE N2,32
1330 RS = RS + 1
1340 LN = NL: GOSUB 11100:C1%(RS)
     = I
1350  FOR I = 0 TO 4
1360 RS%(RS,I) = NUM%(I)
1370  NEXT
1380  RETURN
1999 :
2000  REM TRADUZIONE "UNTIL"
2001 :
2100  IF RS = 0 THEN  PRINT  CHR$
     (7): PRINT "?UNTIL WITHOUT R
     EPEAT ERROR IN ";NL: POP : GOTO
     60000
2110  POKE N1,173
2120  POKE N1 + 1,198
2130  POKE N1 + 2,40
2140 N1 = N2 + 1
2150  IF  PEEK (N1) = 0 THEN 2180
```

Seguito figura 4.

```
2160  POKE N1 - 4, PEEK (N1)
2170 N1 = N1 + 1: GOTO 2150
2180 N1 = N1 - 4
2190  POKE N1,41
2200  POKE N1 + 1,196
2210  IF C1%(RS) <  = 2 THEN  FOR
     I = 0 TO 1: POKE N1 + 2 + I,
     RS%(RS,I): NEXT : GOTO 2300
2220 BN = C1%(RS) - 2
2230 BS = N1 + 4:N3 = II
2240  GOSUB 10000
2245 PI = PI + BN
2250  FOR I = 0 TO C1%(RS) - 1
2260  POKE N1 + 2 + I,RS%(RS,I)
2270  NEXT
2300 RS = RS - 1
2310  RETURN
2999 :
3000  REM TRADUZIONE 'WHILE'
3001 :
3110  POKE N1,173
3120  POKE N1 + 1,198
3130  POKE N1 + 2,40
3140 N1 = N2 + 1
3150  IF  PEEK (N1) = 0 THEN 3180

3160  POKE N1 - 4, PEEK (N1)
3170 N1 = N1 + 1: GOTO 3150
3180 N1 = N1 - 4
3190  POKE N1,41
3200  POKE N1 + 1,196
3210 WS = WS + 1
3220 WS(WS) = N1 + 2:IW(WS) = II
3230 LN = NL: GOSUB 11100
3240  FOR I = 0 TO 4
3250 WS%(WS,I) = NUM%(I)
3260  NEXT
3270  RETURN
3999 :
4000  REM TRADUZIONE 'WEND'
4001 :
4050  IF WS = 0 THEN  PRINT  CHR$
     (7): PRINT "?WEND WITHOUT WH
     ILE ERROR IN ";NL: POP : GOTO
     60000
4100  POKE N1,171
4110  FOR I = 0 TO 4
4120  POKE N1 + 1 + I,WS%(WS,I)
4130  NEXT
4140 PP =  PEEK (PI) + 256 *  PEEK
     (PI + 1): IF PP = 0 THEN 428
     0
4150 LN =  PEEK (PI + 2) + 256 *
     PEEK (PI + 3)
4160  GOSUB 11100
4170 CV = I
4180  IF CV <  = 2 THEN 4280
4190 BN = CV - 2
4200 BS = WS(WS) + 2
4210 N3 = IW(WS)
4220  GOSUB 10000
4230 PI = PI + BN
4240  FOR I = 0 TO CV - 1
4250  POKE WS(WS) + I,NUM%(I)
4260  NEXT
4270  GOTO 4320
4280  IF PP = 0 THEN  POKE WS(WS)
     ,128: POKE WS(WS) + 1,32: GOTO
     4320
4290  FOR I = 0 TO 1
4300  POKE WS(WS) + I,NUM%(I)
4310  NEXT
4320 WS = WS - 1
4330  RETURN
4999 :
5000  REM TRADUZIONE 'IF-THEN'
5001 :
5100  POKE N1,173
5110  POKE N1 + 1,198
5120  POKE N1 + 2,40
5130 N1 = N2 + 1
5150  IF  PEEK (N1) = 0 THEN N1 =
     N1 - 1: GOTO 5175
5160  POKE N1 - 1, PEEK (N1`
5170 N1 = N1 + 1: GOTO 5150
5175 ER = 0: FOR I = 5 TO 0 STEP
     - 1:N1 = N1 - 1: IF  PEEK (
     N1) <  > TH%(I) THEN ER = 1
5180  NEXT : IF ER THEN  PRINT  CHR$
     (7): PRINT "?SYNTAX ERROR IN
     ";NL: POP : GOTO 60000
5190  POKE N1,41
5200  POKE N1 + 1,196
5210 IS = IS + 1
5220 IS(IS) = N1 + 2
5230  RETURN
5999 :
6000  REM TRADUZIONE 'ELSE'
6001 :
6050  IF IS = 0 THEN  PRINT  CHR$
     (7): PRINT "?ELSE WITHOUT IF
     ERROR IN ";NL: POP : GOTO 6
     0000
6100 LN =  PEEK (PI + 2) + 256 *
     PEEK (PI + 3)
6110  GOSUB 11100
6120  FOR I = 0 TO 4
6130  POKE IS(IS) + I,NUM%(I)
6140  NEXT
6150  POKE N1,171
6160 IS(IS) = N1 + 1
6170  RETURN
6999 :
7000  REM TRADUZIONE 'ENDIF'
7001 :
7050  IF IS = 0 THEN  PRINT  CHR$
     (7): PRINT "?ENDIF WITHOUT I
     F ERROR IN ";NL: POP : GOTO
     60000
7100 LN = NL
7110  GOSUB 11100
7120  FOR I = 0 TO 4
7130  POKE IS(IS) + I,NUM%(I)
7140  NEXT
7150  POKE N1,178
7160  POKE N2,32
7170 IS = IS - 1
7180  RETURN
9999 :
10000  REM CREASPAZI
10001 :
10100  FOR I = UB TO BS STEP  - 1

10110  POKE I + BN, PEEK (I)
10120  NEXT
10130 UB = UB + BN: GOSUB 12000
10140 T1 =  PEEK (N3):T2 =  PEEK
     (N3 + 1)
10150  IF T1 = 0 AND T2 = 0 THEN
     RETURN
```

Seguito figura 4.

```
10160 T1 = T1 + BN
10170  IF T1 > 255 THEN T1 = T1 -
      256:T2 = T2 + 1
10180  POKE N3,T1: POKE N3 + 1,T2

10190 N3 = T1 + 256 * T2
10200  GOTO 10140
10999 :
11000  REM CONVERSIONE NUM. LINEA

11001 :
11050 LN =  PEEK (N4) +  PEEK (N4
      + 1) * 256
11100  FOR I = 0 TO 4
11110 NUM%(I) = 32
11120  NEXT
11130 I = 0:SG = 0
11140 DV = 10000
11180 T =  INT (LN / DV):LN = LN -
      T * DV
11190  IF T > 0 OR SG THEN NUM%(I
      ) = T + 48:I = I + 1:SG = 1
11200  IF DV = 1 THEN  RETURN
11210 DV = DV / 10
11220  GOTO 11180
11999 :
12000  REM INCREMENTO PUNTATORI
12001 :
12090  FOR I = 0 TO 4 STEP 2
12100 J =  PEEK (852 + I) + 256 *
      PEEK (853 + I)
12110 J = J + BN
12120  POKE 852 + I,J -  INT (J /
      256) * 256
12130  POKE 853 + I, INT (J / 256
      )
12140  NEXT
12150  POKE 864,UB -  INT (UB / 2
      56) * 256
12160  POKE 865, INT (UB / 256)
12170  RETURN
59999 :
60000  REM RITORNO PROGR.PRINCIP.

60001 :
60010  IF RS > 0 THEN  PRINT  CHR$
      (7): PRINT "UNTIL NOT FOUND
      ERROR"
60020  IF WS > 0 THEN  PRINT  CHR$
      (7): PRINT "WEND NOT FOUND E
      RROR"
60030  IF IS > 0 THEN  PRINT  CHR$
      (7): PRINT "ENDIF NOT FOUND
      ERROR"
60100  CALL 768
```

CASIO

Caccia alla cifra. Un "display-game" per la FX 702P

Una delle caratteristiche più interessanti della FX-702P è la possibilità di gestire il suo display che non è certo paragonabile al video di un personal, ma che per un pocket-computer risulta comunque molto versatile, per creare giochi con simpatici effetti grafici e che richiedono velocità ed attenzione da parte del giocatore.

di **G. De Nicolao**

Il gioco presentato in questo articolo è un esempio semplice, ma divertente, di come si possono ideare dei veri e propri display-game (detti anche video-game dei poveri) con la FX 702P.

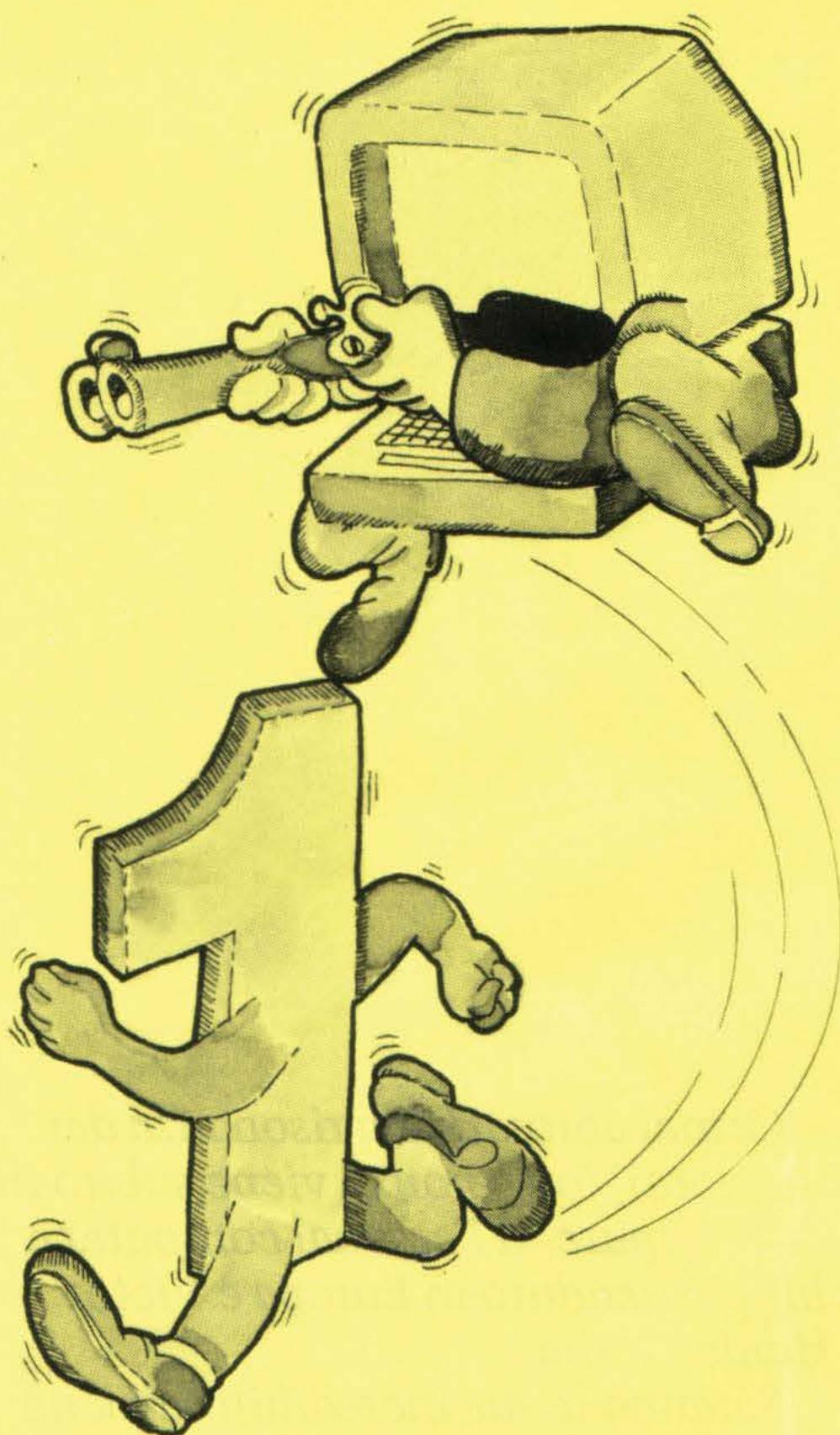
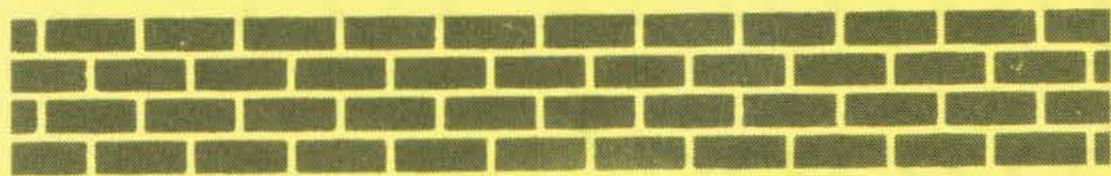
Spieghiamo come si deve procedere per andare a caccia di cifre. Una volta ricopiato il programma in una delle 10 aree di memoria, si può iniziare a giocare premendo RUN EXE. Apparirà subito la scritta "VELOCITA':(1/20)?", a cui il giocatore dovrà rispondere con un numero compreso tra 1 e 20 corrispondente alla velocità desiderata (si tenga presente che con velocità uguali o superiori a 15 è praticamente impossibile vincere, anche se può essere stimolante mettere alla prova la propria resistenza a ritmi così frenetici). Se per esempio avremo premuto "10 EXE", apparirà una seconda scritta: "LIVELLO:(1/7)", cui si deve rispondere con un numero compreso tra 1 e 7. Il livello, come spiegherò più avanti, è direttamente proporzionale alla difficoltà.

A questo punto inizia il gioco vero e proprio; sulla destra del display appariranno i simboli "—0XXXX" e, dopo qualche secondo, apparirà sulla sinistra un numero casuale compreso tra 0 e 9. Se si preme ripetutamente il tasto "+", lo "0" che precede i simboli "XXXX" diventerà un "1", poi un "2", un "3" e così via fino al "9" per riprende nuovamente dallo "0". Lo scopo del gioco consiste nell'incrementare questo contatore mediante il tasto "+" fino a che il suo valore non sia uguale a quello del numero apparso sulla sinistra. A questo punto premendo il tasto "E" (non quello della lettera alfabetica, ma quello vicino al "+", che sta per esponente), se il valore del contatore è uguale al numero mostrato, il numero scomparirà per far posto ad un nuovo numero casuale spostato di uno spazio verso sinistra. Si raggiungerà la vittoria quando il numero casuale sarà stato ricacciato "fuori dal display", cioè all'estrema sinistra del giocatore. Sembrerebbe fin troppo facile, se non ci fosse un piccolo inconveniente: qualora il giocatore sia troppo lento nelle sue operazioni, il numero casuale scompare per lasciare posto ad un nuovo numero spostato di una posizione verso destra. Ogni qual volta il numero raggiunge il contatore, ciò decreta la sconfitta del giocatore.

La velocità scelta all'inizio determina il tempo a disposizione per "cacciare" ciascun numero casuale, cioè il tempo entro il quale bisogna porre il contatore uguale al numero e premere il tasto "E", prima che il numero si sposti verso destra. Il livello scelto all'inizio invece fissa la posizione iniziale del numero casuale: un livello molto alto concede pochi errori, un livello basso al contrario fa partire il numero da lontano e, nel caso di una partenza bruciante da parte del giocatore, permette di ottenere rapide vittorie.
Sia in caso di vittoria che di sconfitta alla fine del gioco compare sul display la scritta "TEMPO=" seguita da un numero proporzionale alla durata dell'incontro: questo parametro può servire a misurare la propria rapidità a vincere e quindi a confrontare l'abilità di due diversi giocatori.

Un consiglio: se invece di premere ripetutamente il tasto "+", lo si mantiene premuto, si ottiene un incremento più rapido del contatore, anche se bisogna essere rapidi nel fermarsi in tempo.

```
LIST
  10 VAC :INP "VELOC
     ITA: (1-20)",D:
     D=21-D:INP "LIV
     ELLO: (1-7)",M
  20 B=M+1:T=0
  30 PRT CSR 13;"-";
     C;"XXXX";
  40 B=B+1:R=INT (RA
     N#*10):PRT CSR
     B;R;
  50 FOR N=1 TO D:T=
     T+1:IF KEY="+";
     GSB 100
  60 IF KEY="E";IF C
     =R;GOTO 200
  70 NEXT N
  80 IF B=9;WAIT 0:P
     RT :PRT "HAI PE
     RSO  TEMPO=";T:
     END
  90 GOTO 40
 100 C=C+1:IF C=10;C
     =0
 110 PRT CSR 14;C;:R
     ET
 200 B=B-1:IF B>0 TH
     EN 220
 210 WAIT 0:PRT :PRT
      "BRAVO! HAI VI
     NTO  TEMPO=";T:
     END
 220 R=INT (RAN#*10)
     :PRT CSR B;R;"
     ";:GOTO 50
```

Figura 1 - Il listato BASIC.

STRUTTURA DEL PROGRAMMA

10	Input della velocità e del livello.
20	Inizializzazione della posizione B del numero casuale e della variabile T che misura il tempo.
30	Costruzione dell'immagine sul display.
40	Creazione del numero casuale R e sua stampa.
50-70	Ciclo la cui lunghezza dipende dalla velocità D durante il quale è lecito incrementare il contatore oppure premere "E".
80	Se la posizione B del numero casuale ha raggiunto l'estremo destro, il giocatore ha perso e viene mostrato per quanto tempo ha resistito.
90	Altrimenti si procede alla creazione del prossimo numero casuale spostato di una posizione a destra.
100-110	Routine di incremento e stampa del contatore.
200	Subroutine che, nel caso sia stato premuto "E" e numero e contatore siano uguali, sposta il numero casuale verso sinistra.
210	Se il numero casuale è stato ricacciato all'estrema sinistra, il giocatore ha vinto e gli viene mostrato il tempo impiegato.
220	Altrimenti viene creato e stampato il prossimo numero casuale R ed il gioco continua.

Statistica ad una dimensione

Con questo numero inizia una breve serie di articoli aventi come argomento dei programmi applicativi di calcolo statistico.
La serie inizia con un programma che consente la rappresentazione grafica o sintetica (cioè attraverso indici statistici), di variabili statistiche ad una dimensione; coprendo, in pratica, la parte corrispondente al primo capitolo di un qualsiasi testo universitario di statistica (diseguaglianza di Tchebycheff esclusa).
Si è ritenuto opportuno, a tale proposito, far precedere alla applicazione pratica una breve trattazione teorica dell'argomento. Coloro che hanno già cognizioni statistiche di base, possono tranquillamente saltare l'articolo su questo numero (a meno che non vogliano effettuare un rapido ripasso!); coloro che, invece, sono completamente digiuni al riguardo, sono vivamente pregati di leggere queste brevi note, in modo da poter apprezzare appieno le potenzialità offerte dal programma.

di **U.G. Barzaghi** **Parte prima**

Il termine "statistica" risale al XVII secolo. Esso fu usato per la prima volta da professori tedeschi di quelle discipline che oggi vanno sotto il nome di scienze politiche, per indicare una materia che doveva essere studiata dai futuri dirigenti e che forniva tutte le informazioni riguardanti lo Stato. In origine, cioè, la statistica non si basava sui dati numerici ma su dati qualitativi.

In seguito, sono convogliate in essa nozioni fondamentali provenienti da altri campi di indagine che ne hanno a poco a poco trasformato la struttura. Ad esempio la "Dottrina della Sorte", antico nome del calcolo delle probabilità; oppure la "Aritmetica Politica", sorta a Londra sotto l'influenza dell'empirismo inglese che, sviluppando il lato numerico della statistica, proponeva un metodo classificatorio dei fenomeni naturalistici relativi alle popolazioni (nascite, morti, censimenti, ecc.); infine hanno contribuito allo sviluppo della statistica tutte quelle discipline che si sono dedicate all'arte del misurare, come la Astronomia, la Topografia e la Geodesia.

Storicamente, la statistica ha cominciato con l'essere descrittiva. Il suo primo impiego è principalmente consistito nell'accumulare informazioni, nell'analizzarle e, essenzialmente, nel sintetizzarle. Solo in seguito, grazie all'apporto del calcolo delle probabilità, si sono evidenziate delle forme generali di legami strutturali che hanno suggerito teorie relative ai fenomeni osservati, poi confermate dai fatti.

Il programma qui presentato si occupa del primo dei due filoni che hanno caratterizzato lo sviluppo storico della materia; vale a dire la "Statistica descrittiva", lasciando lo studio dei modelli probabilistici a sviluppi successivi di una possibile serie di programmi.

La variabile statistica ad una dimensione

Data una popolazione composta da N individui, la statistica insegna come classificarla in relazione ad un unico suo aspetto, quello che interessa l'indagine, e come descriverla o rappresentarla sinteticamente attraverso una serie di indici significativi.

È importante innanzitutto, delimitare con precisione la popolazione che si vuole esaminare. Ciò si ottiene con la definizione della caratteristica comune a tutti i componenti la popolazione in maniera da non consentire ambiguità o indeterminatezze. Una volta definita la popolazione si può esaminarne la struttura, sotto il punto di vista di altre caratteristiche, le quali non si presentano in maniera uniforme per tutti gli individui. Si chiamerà attributo di una popolazione una particolare caratteristica degli individui della popolazione stessa che si manifesta in essi con forme differenti.

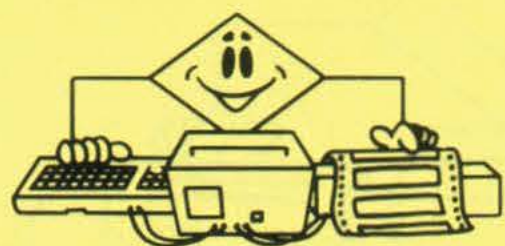
Studiare una popolazione dal punto di vista di un suo attributo significa classificare gli individui che compongono la popolazione secondo le varie forme dell'attributo. Per poter far ciò occorre che l'attributo X in relazione al quale si vuole effettuare la organizzazione della popolazione, possegga i seguenti requisiti:

a) Che l'attributo possa assumere forme distinte Xi tra loro incompatibili; ogni individuo deve poter possedere solo una forma dell'attributo.

b) Che ogni forma Xi rappresenti un concetto di classe; deve essere possibile cioè, almeno in linea di principio, che parecchi individui posseggano la stessa forma dell'attributo.

c) Che in ogni individuo della popolazione sia presente una delle forme Xi; l'attributo deve essere presente, pur sotto diverse forme, in ciascun individuo della popolazione.

d) Che esistano almeno due individui nella popolazione in possesso di forme diverse dell'attributo, e cioè che la popolazione stessa non si presenti totalmente omogenea.

La successione delle diverse forme Xi non sempre può rappresentarsi con dei numeri in maniera immediata, inoltre, l'argomento X può assumere o un numero finito di forme, e in tal caso è detto discreto; oppure può presentarsi con un numero infinito di forme, tutte però contenute con continuità in un intervallo limitato, ed allora è detto di

tipo continuo.
Il programma in questione prende in considerazione solo il caso di argomenti discreti, per cui la successione dei valori argomentali di X può rappresentarsi con una successione di Xi (i = 1, 2, ..., n).
Per mezzo dell'operazione di classificazione secondo l'argomento X degli N individui che formano nel loro insieme la popolazione C, si raggruppano assieme quegli individui che posseggono una uguale forma dell'attributo. Quindi si contano gli individui in ciascun gruppo ottenendo i valori F1, F2, ..., Fn della frequenza assoluta. Essi rappresentano cioè il numero degli individui compresi nelle classi definite dai valori argomentali corrispondenti X1, X2, ..., Xn.
Le stesse classi possono essere rappresentate diversamente sostituendo alla successione delle frequenze assolute, la successione delle frequenze relative Fi/N. La X, costruita come si è detto e rappresentata in una delle due forme, prende il nome di "variabile statistica ad una dimensione".
In alcuni casi la classificazione di una popolazione rispetto ad un determinato valore argomentale può essere fatta usando non più le singole forme dell'attributo in questione, ma delle "classi" di opportuna ampiezza. Ad esempio, si può determinare il numero di individui il cui valore argomentale è compreso tra Xi e Xj, estremi inclusi, e associare alla classe Xi — Xj tale numero, che verrà quindi chiamato frequenza assoluta della classe. Ogni classe è inoltre contraddistinta da un valore medio o punto medio della classe dato da (Xi+Xj)/2.

Rappresentazione grafica di una variabile statistica semplice

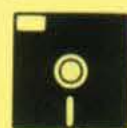
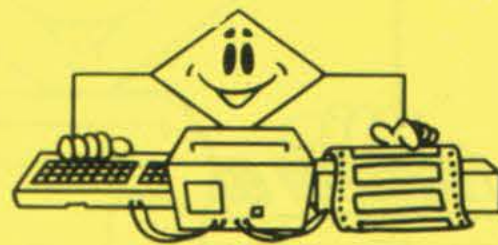
Della X si possono dare alcuni tipi di rappresentazioni grafiche. Su di una semiretta, si riportano, in una scala opportuna, i valori numerici dell'argomento X, dopo averli disposti in ordine crescente.
In corrispondenza a ciascun valore Xi si riporta, in direzione normale alla semiretta, un segmento di lunghezza proporzionale alla frequenza assoluta o relativa di quel valore argomentale. La figura risultante si chiama "istogramma".
Può capitare che l'ampiezza delle classi coincida con l'unità di misura delle x e che venga mantenuta costante. In generale, però, questo non accade ed anzi è opportuno modificare l'ampiezza delle classi in quella zona in cui le frequenze diminuiscono sensibilmente. Nella costruzione degli istogrammi ha importanza notevole la scelta dell'ampiezza dell'intervallo di ciascuna classe. Da questa scelta dipende a volte in modo assai macroscopico la formazione dell'istogramma. L'ampiezza dell'intervallo dovrà essere scelta, sia pure con una certa arbitrarietà, tenendo conto di questi elementi fondamentali:
- Dell'estensione del campo di variabilità dell'argomento.
- Del numero complessivo di individui componenti la popolazione.
- Del grado di attendibilità delle rilevazioni dell'argomento stesso.

Nel caso in cui il valore dell'argomento proprio di ciascun individuo, ad esempio, sia ottenuto mediante misure, sarà inutile ripartire l'istogramma in classi il cui intervallo sia inferiore alla sensibilità della stessa misura. La rappresentazione mediante istogramma con intervallo di ampiezza uniforme della variabile statistica già ordinata in classi, cambia sensibilmente la sua forma corrispondente al variare dell'ampiezza dell'intervallo.
Il problema della scelta dell'intervallo in cui suddividere la scala dei valori argomentali per la costruzione dell'istogramma perde la sua importanza qualora si passi dalla rappresentazione della variabile statistica a quella della variabile statistica cumulata.
Si dice "funzione cumulativa di frequenza" o "funzione di distribuzione" una doppia successione in corrispondenza biunivoca in cui in corrispondenza di ogni valore Xi dell'argomento, viene posta la somma delle frequenze di tutti gli elementi che hanno valori inferiori od uguali a Xi.

Moda, mediana, quantili

Nelle serie dei valori argomentali ve ne sono alcuni di particolare importanza per la descrizione della variabile statistica a cui si riferiscono. Esse sono:
a) La "moda", ossia quel valore argomentale a cui corrisponde la massima frequenza, rispetto ai valori argomentali che lo precedono e lo seguono: pertanto, il valore modale non può mai occupare il primo o l'ultimo posto nella successione dei valori argomentali Xi. Se il valore di massima frequenza coincide con il primo o l'ultimo termine, la distribuzione si dice "zeromodale". Una distribuzione può inoltre essere "unimodale" o "plurimodale" a seconda che vi sia una sola frequenza che supera tutte le altre, oppure che vi siano nella serie delle frequenze altri massimi relativi.
b) La "mediana", ossia quel valore argomentale la cui frequenza Fm sia tale che:
1) la somma delle frequenze che precedono Fm è minore della metà della frequenza totale;
2) la somma di Fm e di quelle frequenze che la precedono è uguale o maggiore della metà della frequenza totale.
Nell'istogramma della variabile suddetta, la mediana è l'ascissa del punto la cui ordinata divide in due parti uguali l'area coperta dall'istogramma.
c) I "quantili", ossia quei valori argomentali per cui sono soddisfatte condizioni analoghe a quelle poste per la mediana, sostituendo soltanto "un quarto" o "tre quarti" al posto di "metà" nella condizione b. 1) suddetta.

Rappresentazione sintetica di una variabile statistica a una dimensione

La rappresentazione grafica della variabile statistica nelle forme sopra viste, pur essendo spesso assai espressiva da un punto di vista qualitativo, non permette però di ricavare molte indicazioni quantitative sulla distribuzione. Per ottenere una descrizione matematica più accurata sono stati introdotti alcuni indici che, ricavati usando tutti i dati che compongono la variabile, ne precisano le caratteristiche essenziali. Tali indici descrittivi vengono nel loro complesso chiamati "statistiche".
È necessario introdurre alcune definizioni: si dice momento k-esimo rispetto al polo theta, di

una distribuzione di frequenza o variabile statistica a una dimensione, la sommatoria del prodotto della differenza dei valori degli argomenti rispetto al polo theta, elevata ad esponente k, moltiplicata per la frequenza relativa associata al valore argomentale suddetto. La serie dei momenti di una variabile statistica, rispetto ad un particolare valore del polo, per esempio lo zero, basta a rappresentare tutte le caratteristiche della sua distribuzione.

I momenti che nella pratica sono più significativi per rappresentare la funzione sono i tre seguenti:
a) Il momento di primo ordine (k=1) della variabile rispetto al polo theta = 0, che prende il nome di media della variabile statistica.
b) Il momento di secondo ordine (k=2) della variabile rispetto al polo theta = 0, che prende il nome di "valore quadratico medio".
c) Il momento di secondo ordine (k=2) della variabile rispetto alla media, che prende il nome di "varianza" o "momento centrale del secondo ordine". La radice quadrata della varianza, prende il nome di "scarto quadratico medio" o "deviazione standard".

Vengono inoltre introdotti due indici che consentono di misurare il grado di asimmetria di una distribuzione:
a) Indice di skewness - dato dalla formula (Media-Moda)/(scarto quadratico medio).
b) Coefficiente gamma - dato dal momento di terzo ordine rispetto alla media, diviso il cubo dello scarto quadratico medio.

Il coefficiente gamma risulta nullo nel caso in cui la distribuzione sia simmetrica, positivo se il diagramma di frequenza presenta una "coda" verso destra e negativo nel caso contrario. L'uso degli indici di asimmetria va fatto tuttavia sempre con discernimento perché possono verificarsi casi in cui, ad esempio, si annulla il momento del terzo ordine, e quindi il "coefficiente gamma", senza che la variabile statistica sia simmetrica.

Alcuni significati collaterali degli indici statistici

Ai due momenti chiamati media e varianza è associato un importantissimo significato descrittivo della variabile statistica X. La media è un indice di posizione, cioè determina la posizione della popolazione cui la variabile statistica si riferisce, sulla scala dei valori argomentali. Per comprendere il significato di media può essere utile una analogia meccanica. Se i valori della frequenza si interpretano come i valori di altrettante masse disposte lungo un asse ed aventi come coordinata il valore dell'argomento corrispondente, la media, coincide con il baricentro delle masse.

La varianza è un indice di variabilità, cioè essa misura la dispersione dei valori dell'argomento posseduti dagli individui, intorno al valore medio. Il concetto di varianza è fondamentale in statistica; la statistica, anzi, rappresenta proprio un metodo per studiare la variabilità di una caratteristica in una popolazione di individui. Storicamente si è giunti alla scelta di questo indice a causa della necessità di misurare la variabilità esistente fra i dati, mettendo a confronto le distanze fra ciascun individuo della popolazione ed un termine fisso.

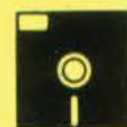
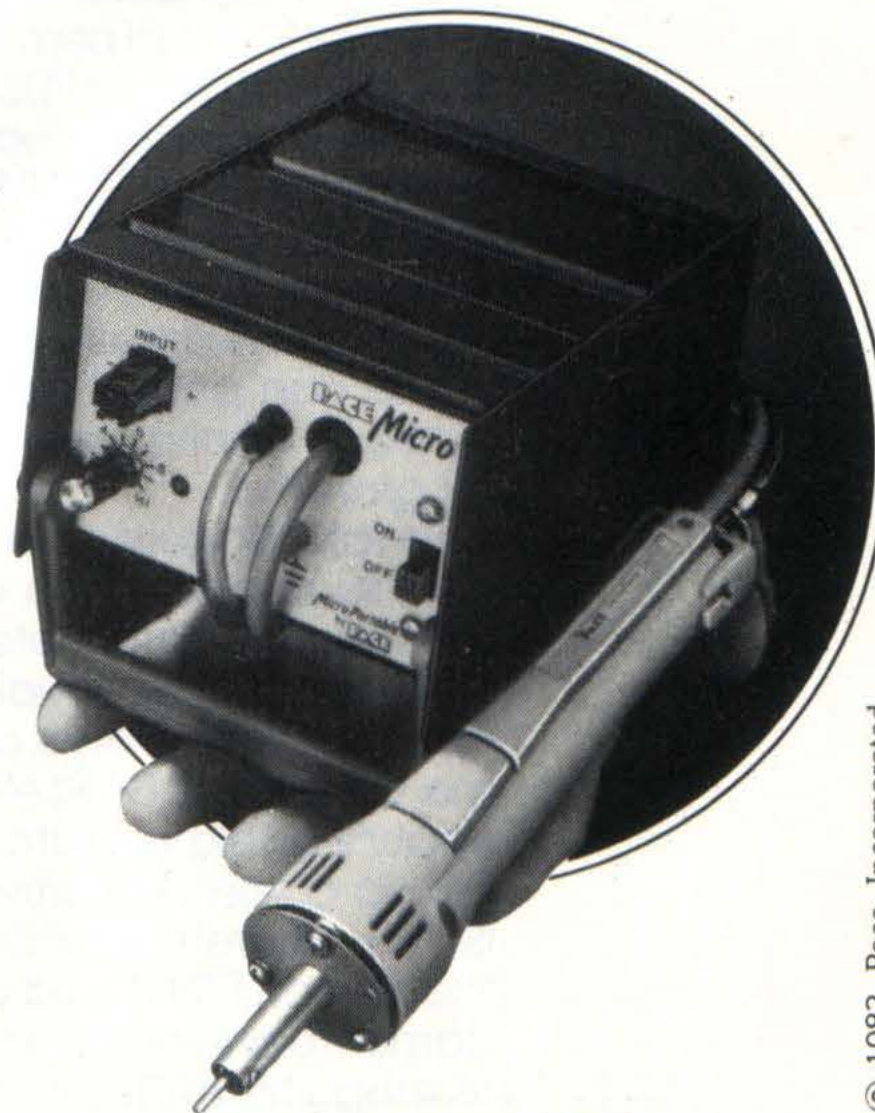
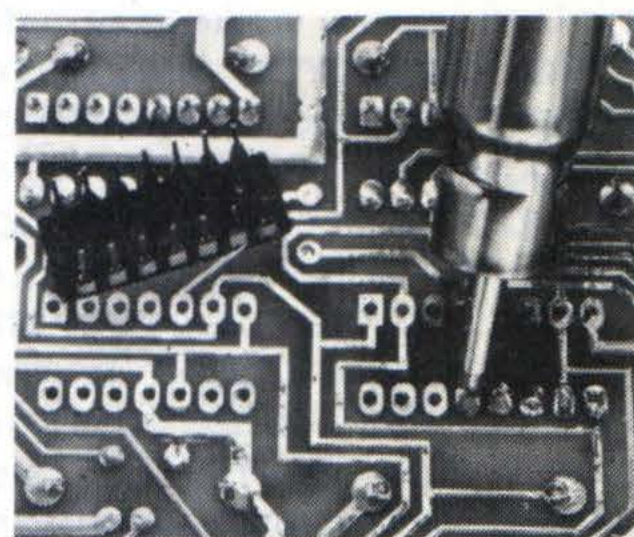
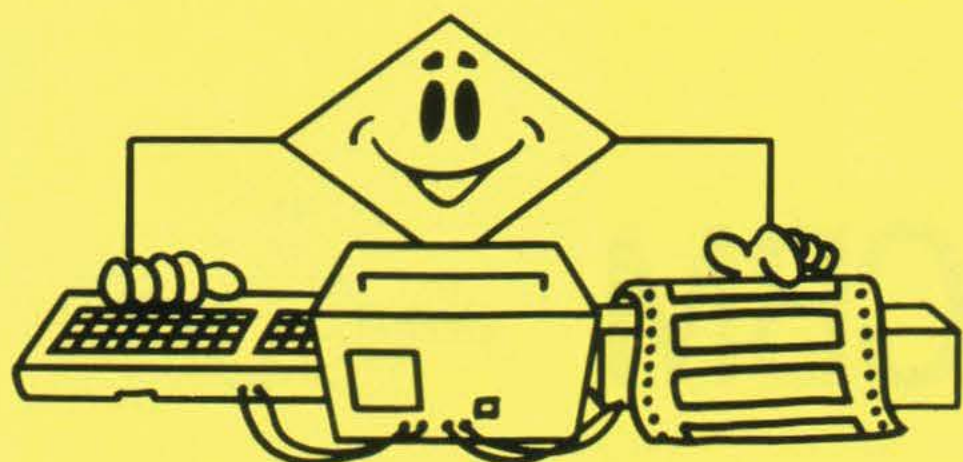
L'analogia meccanica che meglio si presta per aiutare a comprendere il significato di varianza è quella del momento di inerzia, a patto di interpretare le frequenze relative come masse e le differenze tra i valori degli argomenti e la media, come le distanze delle masse dal centro di rotazione, si può interpretare la varianza come rotore, cioè quella distanza dall'asse di rotazione in cui si potrebbe concentrare tutta la massa per conservare invariato il momento di inerzia.

Dalla definizione meccanica di momento di inerzia, risulta che, a parità di massa, esso è tanto maggiore quanto maggiori sono le distanze dall'asse.

Per ogni riferimento vedi: "Fondamenti di Statistica", della prof. Giovanna Togliatti, edito dalla Hoepli.

Epilogo

Queste note sono più che sufficienti alla comprensione dell'argomento trattato. Sul prossimo numero, che conterrà anche il listato del programma, debitamente commentato, vedremo il funzionamento del programma stesso, le sue opzioni, i vari formati in cui i dati possono essere introdotti e le rispettive limitazioni.

(Continua)

Util 75: dalla parte del programmatore

Chiunque legga questa rivista è, chi più chi meno, un programmatore. Ecco dunque un articolo, preparato soprattutto per l'HP-75C e la stampante HP 82905B, fatto di piccole cose, che forse saranno già note a qualcuno, ma che potranno tornare utili ai più, quando si presenti il caso.

di **V. Anselmo**

Il programma Ricupero permette di stampare il listato di un programma anche quando, per varie operazioni di riduzione al minimo, questo non sarebbe più stampabile.

Tanto per cominciare, esploriamo le possibilità di controllare da programma il cursore. Proviamo ad immettere ed eseguire il programma di figura 1.

Per poter ottenere facilmente le parole sottolineate che compaiono nel programma, basterà un programmino di poche righe, come riportato in figura 2.

Dopo aver eseguito il suddetto programmino la parola sottolineata si otterrà premendo CTL 1.

Ma veniamo al primo programma. Eseguendolo si vedrà la parola PLUS avvicinarsi e fondersi con SOFTWARE fino a creare SOFTPLUS, quindi PLUS lampeggerà su SOFTWARE fermandosi poi con SOFTPLUS. La stringa D$ contiene 11 spazi vuoti e nella riga 20, tra A$ e B$ ci sono 5 spazi. Il lettore potrà anche provare a far posizionare, per esercizio, le due parole agli estremi del visore e farle riunire al centro, con un'opportuna modifica di programma.

Se fate leggere un archivio di testo visualizzandolo su uno schermo con l'interfaccia video HP 82163A collegata, noterete che c'è una differenza in velocità di visualizzazione a seconda che abbiate usato DISP e assegnato l'interfaccia come dispositivo DISPLAY IS (con DISPLAY IS': TV'), oppure abbiate usato PRINT e assegnato l'interfaccia come dispositivo PRINTER IS (con PRINTER IS':TV'). Vedrete che un testo mandato all'interfaccia con PRINT sarà visualizzato molto più rapidamente che non con DISP, anche se prima avete eseguito un DELAY 0. Provate a far visualizzare un'intera pagina di testo e vedrete la differenza.

Passando alla stampante, sappiamo che, quando si stampa una riga composta di tante parti legate dal punto e virgola (PRINT A$; @ PRINT B$; ecc.), la stampante HP 82905B stampa la sua riga solo quando nel buffer di stampa è stata accumulata tutta una riga (quando arriva l'ottantunesimo carattere in modo normale), o quando si manda un PRINT senza il punto e virgola finale. Anche il visore dell'HP-75C ci sembra comportarsi nello stesso modo, perchè siamo abituati a vedere comparire la stringa sempre di colpo, ma non è vero. Sul visore le varie parti della stringa da visualizzare (con DISP A$; @ DISP B$; ecc.) vengono visualizzate man mano che sono trasmesse. Basterà un programmino come quello in figura 3 per dimostrarlo.

Sul visore i caratteri vengono presentati l'uno dietro l'altro con un breve intervallo, mentre la stampante stamperà la riga solo alla fine, quando incontra il PRINT finale. Sulla visualizzazione immediata nel visore delle parti seguite dal punto e virgola non ha alcuna influenza il DELAY. Si provi a dare un DELAY 5 prima dell'esecuzione, e poi un DELAY 0: sarà la stessa cosa.

Ad accumulare i dati da stampare e a presentarli tutti assieme è il buffer della stampante. Basterà infatti avviare il programma di figura 4 per accertarcene.

Sul visore, che ora sostituisce la stampante, i caratteri appariranno uno dopo l'altro, intervallati da un'attesa di un decimo di secondo.

La stampante stamperà la riga solo quando questa è completa, anche se sarà stata assegnata come dispositivo DISPLAY IS (con DISPLAY IS': P') e indirizzata con DISP, sempre per lo stesso motivo (buffer di accumulo nella stampante).

Verrà stampata solo la riga completa anche quando avremo mandato dei PRINT senza il punto e virgola dopo aver predisposto la stampante per la stampa continua (senza ritorno carrello/interlinea) con ENDLINE". Provare con il programma di figura 5 che dimostra come in tal caso la stampa si ottiene solo quando si manda un PRINT dopo aver riportato le cose alla normalità con ENDLINE da solo, o quando il buffer della stampante supera la capacità massima della riga in accordo con il tipo di carattere adottato (normale, compresso, ecc.). Un'istruzione PRINT quando ci si trova in condizione di ENDLINE" equivale ad un'istruzione PRINT seguita da punto e virgola in condizione ENDLINE CHR$(13)&CHR$(10). In entrambi i casi la sequenza di ritorno carrello/interlinea non viene mandata.

Quando la stampante è nel modo stampa continua non ha alcuna influenza il comando PWIDTH. Provare con PWIDTH 10 e poi con PWIDTH INF: il risultato è lo stesso.

Si riuscirà a far stampare un carattere alla volta sulla stessa riga con un certo intervallo fra l'uno e l'altro solo facendo passare la stampante in modo grafico dopo la stampa di ogni carattere. Provare con il programma in figura 6.

```
10 A$='SOFTWARE' @ B$='PLUS' @ C$='SOFTPLUS' @ D$='
20 DELAY 0 @ DISP D$&A$&'        '&B$ @ WAIT .5
30 FOR J=25 TO 15 STEP -1
40 DISP D$&A$;CHR$(27);'%';CHR$(J);CHR$(0);B$;' '
50 NEXT J @ WAIT .5
60 FOR J=1 TO 5 @ DISP D$&A$ @ WAIT .1
70 DISP D$&C$ @ WAIT .1 @ NEXT J
```

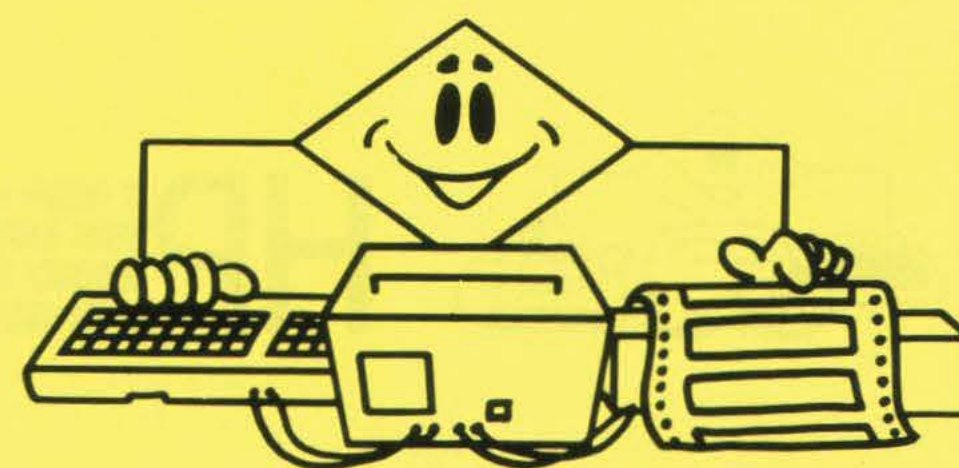



Figura 1 - Il programma per il controllo del cursore.

```
10 DIM B$[50]
20 INPUT 'parola da sottol.(CTL 1): '; A$ @ B$="" @ FOR I=1 TO LEN(A$)
30 B$=B$&'_('&CHR$(NUM(A$[I,I])+128) @ NEXT I
40 DEF KEY '1',B$;
```

Figura 2 - Un programmino di sottolineatura.

```
10 FOR I=65 TO 90
20 DISP CHR$(I); @ PRINT CHR$(I); @ WAIT .1
30 NEXT I @ PRINT @ DISP
```

Figura 3 - Una dimostrazione che le parti di stringa visualizzate sul visore vengono trasmesse singolarmente.

```
10 OFF IO
20 FOR I=65 TO 90
30 PRINT CHR$(I); @ WAIT .1
40 NEXT I @ PRINT
50 RESTORE IO
```

Figura 4 - Il programma che riguarda il buffer della stampante.

```
10 A=0 @ DELAY 2 @ ENDLINE '' @ FOR I=40 TO 79
20 PRINT CHR$(I) @ NEXT I @ DISP '40 caratteri nel buffer' @ GOSUB 110
30 ENDLINE '' @ FOR I=40 TO 119
40 PRINT CHR$(I) @ NEXT I @ DISP '80 caratt. nel buffer: ';
50 IF A THEN DISP 'compresso' ELSE DISP 'normale'
60 DISP 'aggiungo un altro carattere' @ PRINT ' '
70 IF A THEN 90
80 PRINT CHR$(27)&'&k2S' @ A=1 @ GOTO 30
90 DISP 'fine: buffer non scaricato' @ GOSUB 110
100 PRINT CHR$(27)&'&k0S' @ DISP @ END
110 DISP 'scarico con ENDLINE @ PRINT' @ ENDLINE @ PRINT @ RETURN
```

Figura 5 - La stampa si ottiene solamente quando sono soddisfatte le condizioni dette nell'articolo.

```
10 FOR I=65 TO 90
20 PRINT CHR$(I)&CHR$(27)&'*b0G'; @ WAIT 1
30 NEXT I @ PRINT
```

Figura 6 - Passaggio in modo grafico dopo ogni carattere.

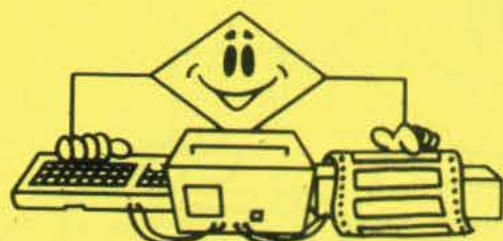



Figura 7 - Differenti tipi di stampa si ottengono agendo sulle spaziature.

```
10 PRINT CHR$(27)&'&l72D'&CHR$(27)&'&k1S'
20 FOR A=1 TO 2
30 FOR I=65 TO 90
40 PRINT CHR$(I);
50 NEXT I @ PRINT
60 NEXT A
70 PRINT CHR$(27)&'&l6D'&CHR$(27)&'&k0S'
```

Figura 8 - Il numero del carattere con sottolineatura spessa.

La sequenza CHR$(27)&'★b0G' fa passare la stampante nel modo grafico dopo la stampa della lettera e la fa ritornare immediatamente nel modo normale senza aver stampato alcun disegno in modo grafico (o meglio dopo aver stampato un disegno nullo). Avrà però fatto scaricare il buffer di stampa ogni volta e la testina avrà così stampato una lettera alla volta.

Un tipo di stampa non previsto normalmente dalla stampante si può ottenere agendo sulla spaziatura fra le righe. Spaziando una volta di un solo punto e facendo ribattere la riga si ottiene una specie di grassetto, più apprezzabile con i caratteri espansi o compressi-espansi e le lettere maiuscole (vedi figura 7).

Uno degli scopi del programmare è quello di ridurre la lunghezza del programma dopo averlo scritto. Il fine è quello di "lasciare ogni byte possibile di memoria all'utente". Programmi brevi, veloci e potenti, quindi. Vi sono vari modi per raggiungere lo scopo. Innanzitutto sarà bene eliminare qualunque giro vizioso e snellire al massimo il programma, togliendo tutto ciò che è inutile.

Dopo aver operato in questo modo più radicale ottenendo dei risparmi a volte inaspettati (come quando, ad esempio, si riesce a sostituire una lunga sequenza di IF .... THEN ... con una semplice formula matematica), si potrà passare ad un lavoro di riduzione più in dettaglio, lavorando su fattori meno importanti, che però riusciranno a far risparmiare talvolta un numero inaspettato di byte.

Per risparmiare memoria e rendere anche il programma un poco più veloce è bene collegare, tutte le volte che è possibile, più istruzioni con il segno @.

Naturalmente non si potrà collegare un'istruzione con IF ... THEN ... alla successiva, o un'istruzione GOTO ... con quella che la segue. Per ogni istruzione che si riesce a mettere con @ sulla stessa riga (in modo legale) si risparmiano 3 byte di memoria.

Dimensionare le variabili all'inizio di un programma è un altro modo di risparmiare qualche byte. Ma non sarà conveniente dimensionare una variabile la cui lunghezza sia prossima a quella presunta dal computer, perchè anche il dimensionamento occupa memoria. Non converrà dimensionare con DIM A$(30) una variabile A$ che dovrà contenere solo due byte in meno del dimensionamento standard dato dal computer (32 byte). L'istruzione per il dimensionamento occuperà infatti 13 byte, facendoci perdere 11 byte.

Un archivio di testo usa meno memoria di un archivio di dati. Quando non si devono memorizzare variabili numeriche, si risparmiano 3 byte per riga di testo se si usa un archivio di testo (TEXT) anzichè un archivio di dati (BASIC).

Quando in un programma BASIC dobbiamo immettere un carattere, si risparmierà memoria se il carattere si immette nella sua forma più breve (ad esempio, DISP 'A' anzichè DISP CHR$(65)). Si risparmieranno due soli byte per il primo carattere, ma cinque per ogni carattere successivo della stessa stringa. Per immettere certi caratteri non direttamente accessibili da tastiera, come i caratteri con numero ASCII dal 144 al 159, dal 183 al 191, dal 207 al 223 e dal 240 al 255, sarà necessario prima ridefinire un tasto facendogli rappresentare quel carattere. Si ricordi che, siccome si tratta di caratteri sottolineati, la ridefinizione dovrà avere sempre la forma:

```
DEF KEY '...',' ('&CHR$(...);
```

con il carattere da rappresentare preceduto dal segno (ottenuto premendo due volte SHIFT I/R. Tra parentesi, si sarà notato che abbiamo messo nella lista anche i caratteri coi numeri 183 (7), 184 (8) e 185 (9), non ottenibili direttamente da tastiera, contrariamente a quanto risulterebbe invece nel prospetto di pagina 290 del manuale operativo dell'HP-75C.

Una riduzione di questo tipo è utile e sempre necessaria su un programma finale, cioè nella versione che deve essere effettivamente usata. Sorgono però dei problemi. Quando si riduce il programma in tal modo, la sostituzione di certi caratteri inizialmente nella forma CHR$ (...) con la loro forma più breve può rendere poi il programma non più stampabile con la stampante HP 82905B, o può addirittura rendere non più visualizzabile sul visore (e sul video, se collegato) un'intera riga di istruzioni.

Riducendo, per esempio il programma seguente con la sostituzione della forma CHR$(27) con la forma più breve (ESC) ottenibile con CTL BACK,

non si riuscirà più né ad osservare, né a far stampare il listato:

```
10 FOR I=1 TO 10@DISP 'I=';I@WAIT 1@DISP
   CHR$(27)& 'E'@NEXT I
```

Dopo la riduzione, che ci avrà fatto guadagnare 5 byte, tutto quello che vedremo e che riusciremo a far stampare della riga 10 del programma sarà:

```
'@ NEXT I
```

perchè tutto il resto sarà stato annullato dal comando di cancellazione dello schermo, che agisce sul visore, sull'interfaccia video e sulla stampante. In casi estremi di una riga può scomparire praticamente tutto. Così con la riduzione di:

```
10 DATA CHR$(8)&CHR$(10)&CHR$(13)
```

avremo, è vero, guadagnato ben 22 byte, ma la riga non verrà più visualizzata. Una riduzione del genere non è neppure facile. Ammesso che il programma da ridurre si chiami PROVA, per ottenere più agevolmente la suddetta riduzione dovremo ricorrere ad un altro programma, come ad esempio:

```
10 ASSIGN#1 TO 'PROVA'
20 PRINT#1, 10: CHR$(8)&CHR$(10)&CHR$(13)
```

Se proveremo ora a visualizzare l'istruzione 10 di PROVA, non ne otterremo nient'altro che un apostrofo (ma potremo notare che il cursore si può spostare a destra di 11 spazi). I tre caratteri sono stati però tutti e tre memorizzati, come potremo facilmente provare facendoli leggere e facendone visualizzare il numero ASCII.

In altri casi, come quando la sostituzione dà luogo a caratteri sottolineati, la riga sarà ancora visualizzata sul visore dell'HP-75C (e sul video se collegato), ma il carattere sottolineato sul visore (e in negativo sullo schermo) non verrà stampato, oppure verrà stampato un carattere diverso (non c'è nessun accordo tra quello che compare sul visore del computer e i caratteri del set alternativo della stampante HP 82905B).

Di solito per ovviare a questo inconveniente si mantengono di uno stesso programma due copie distinte, una con tutti i commenti e tutti i comandi in chiaro (nella forma stampabile, con CHR$(...)) e l'altra ridotta al massimo. Può però capitare, per la fretta, per le molte revisioni successive di un programma, per una momentanea mancanza di supporti di memorizzazione di massa (schedine magnetiche, nastri, dischi) di non poter essere in grado di memorizzare a parte la versione completa più aggiornata. Ci troviamo così spesso con un programma che, a distanza di tempo, resta totalmente illeggibile in certe parti, e non stampabile in modo integrale.

Modificando i programmi MATRICI e STAMPA e integrando l'archivio di dati EXTRA presentati la volta scorsa, potremo ottenere il modo di far stampare dall'HP 82905B anche un programma non più normalmente leggibile. Vediamo come ciò sia possibile.

Nell'archivio EXTRA presentato la volta scorsa mancavano le matrici per i caratteri con i numeri ASCII 8, 10, 13 e 27, perchè questi caratteri non vengono rappresentati sul visore dell'HP-75C. Per far stampare in forma leggibile un programma BASIC, in cui tutti i 256 caratteri possono essere presenti, sarà però necessario aggiungere a quell'archivio anche le quattro matrici mancanti. La prima modifica che si dovrà effettuare riguarda il programma MATRICI. È una modifica necessaria per poter far accettare anche quei 4 caratteri che sono piuttosto difficili da immettere. Modificando la riga 30 di MATRICI nel modo presentato qui sotto, si potranno immettere i caratteri con il loro numero ASCII, anzichè con la loro forma normale:

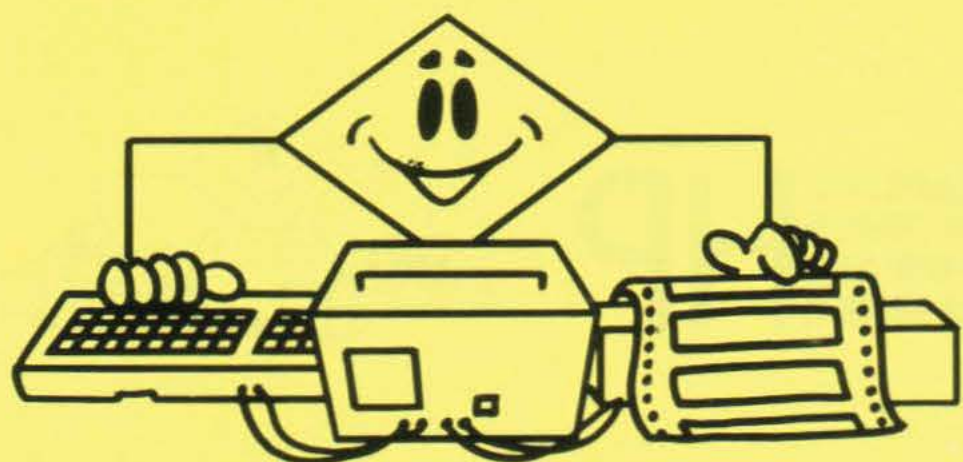
```
30 INPUT 'num. car. corrispondente:';Z@B$=
   CHR$(Z)
```

La forma dei quattro caratteri potrà essere scelta a piacere, purchè rispecchi la realtà del carattere. Qui proponiamo le forme di figura 8 che rappresentano il numero del carattere con una spessa sottolineatura.

Quando ad EXTRA avremo aggiunto anche queste 4 matrici sarà bene cambiargli il nome chiamandolo EXTRA2 per non confonderlo con l'originale da usare con il programma STAMPA. La modifica da apportare a MATRICI era temporanea e, dopo l'uso, potremo cancellarlo dalla memoria.

Le modifiche da apportare a STAMPA invece sono più sostanziali. Il programma dovrà anche prendere un altro nome (RICUPERO) perchè servirà ad uno scopo ben preciso, e non semplicemente a far stampare un archivio di testo così com'è. Per semplificare, si è riportata la nuova versione con tutte le correzioni e con la rinumerazione effettuata. I numeri di riga citati appresso si riferiscono al programma RICUPERO e non a STAMPA. Con l'aggiunta di D$ =' ' nella riga 50, l'aggiunta della riga 60 e dell'ultima parte della riga 150 (IF....GOTO 70) si è provveduto a far stampare rientrate le righe che superavano la lunghezza di stampa prevista dalla stampante in modo normale (80 caratteri). Il numero di riga dell'istruzione risulta così sempre in fuori, sulla sinistra, aumentando la leggibilità. Nelle righe 30 e 40 è stata eliminata la distinzione BASIC/TEXT per gli archivi richiamati, in quanto sia l'archivio da stampare, che l'archivio EXTRA2, sono entrambi BASIC. Si è anche voluto risparmiare qualche byte modificando le istruzioni 80, 90 e 100, abbreviando il messaggio iniziale, e compattando un pò di più il tutto.

La trasformazione del programma BASIC in un archivio di testo si è resa necessaria perchè altrimenti non sarebbe stato possibile leggerlo e stamparlo come testo (avremmo dovuto ricorrere a PLIST).

L'istruzione PWIDTH INF della riga 40, già presente nel precedente programma STAMPA, serve a far sì che la stampante possa stampare in modo grafico superando le eventuali limitazioni di una precedente definizione PWIDTH. L'istruzione PWIDTH controlla infatti il numero di byte inviati alla stampante, e non il numero di caratteri inviati. Ogni carattere speciale stampato in modo grafico equivale a undici byte inviati alla stampante (5 byte di comandi e 6 di testo), anche se quando è stampato sulla carta è largo esattamente come un carattere normale, che vale un byte soltanto.

M 20

Parola di elaboratore (Oulipoit & Patafisica)

Pubblichiamo su R.P. i principali programmi, preceduti da quello principale di pilotaggio, relativi al package poetico-informatico Oulipoit di cui già si è parlato su **Bit** *(n. 43/sett. 83: "Oulipoit, la macchina per poetare"). Se ne fanno carico quattro degli autori, i più giovani per l'esattezza, studenti dell'appena iniziato terzo anno del corso di laurea in Scienza dell'Informazione della Statale di Milano, limitandosi a note essenziali sui diversi listati. Ad eccezione del primo, che inevitabilmente fa riferimento a peculiarità del sistema operativo dell'elaboratore M20 Olivetti, che comunque vengono in parte ricordate, altri sono quasi interamente redatti in BASIC Microsoft, quasi uno standard de-facto che perciò ne consentirà il riciclaggio, a quanti vorranno divertirsi con simili ludi letterario-casuali, nelle macchine più varie, portatili incluse.*
Per non appesantire la presentazione, dividiamo il lavoro in due puntate. Nel frattempo, chi lo desidera, cominci a digitare ...

di **P. Badà, P. Brotzu, P. Ferrara, G. Foglia**

Parte prima

I programmi che costituiscono il package Oulipoit sono stati a disposizione dei visitatori della mostra "Jarry e la Patafisica" per circa tre mesi, sei giorni alla settimana, sette ore al giorno su tre differenti M20. Quando la mostra ha chiuso i battenti, il numero totale delle selezioni operate su Oulipoit e automaticamente registrate su un opportuno file di statistica, ammontava a più di 12.000.
Ora, anche se questo numero è andato al di là delle aspettative, fin dall'inizio della progettazione, era stato curato in modo particolare l'interfaccia utente: ogni visitatore, anche chi non avesse mai visto una macchina per scrivere, doveva essere in grado di portarsi a casa un tabulato con i saluti di Oulipoit.
Inoltre il tutto avrebbe dovuto essere accuratamente protetto contro i tentativi, involontari o meno, di "aprirlo per vedere come è fatto".
Tali problemi sono notoriamente molto difficili e formano oggetto di attenti studi da parte dei maggiori esperti del settore, ma senza falsa modestia possiamo affermare che Oulipoit, per tutto il tempo della sua avventura patafisica, ha resistito abbastanza stoicamente ad angherie di ogni genere capitolando solo di fronte a sabotatori di professione.
Il programma PILOTA costituisce l'ossatura di Oulipoit, e i suoi compiti sono di due diverse nature: estetici e funzionali. Innanzitutto deve presentare l'insieme in modo curioso, così da attrarre l'attenzione di un pubblico alle volte piuttosto diffidente e da invitarlo a "premere qualche tasto"; inoltre ha la funzione di legame tra tutti gli altri programmi e, per questo motivo, si può pensarlo in un certo senso ramificato anche al loro interno.
Il programma è caricato in un file chiamato "init. bas", del tutto particolare: all'accensione del calcolatore, una volta eseguiti i programmi di diagnostica, il sistema operativo dell'M20 (il PCOS) si preoccupa di verificare l'esistenza o meno di un file con questo nome e, in caso positivo, il programma contenuto in "init. bas" viene immediatamente posto in esecuzione in ambiente BASIC, senza nessun intervento da parte dell'operatore. L'accorgimento permette l'avvio automatico del tutto all'accensione.
La prima parte del programma svolge compiti di assestamento dell'ambiente di lavoro di Oulipoit; grazie all'uso del comando CALL (se sono presenti parametri) o del comando EXEC vengono caricate in memoria le seguenti routine di sistema:
- "sp" che permette di ottenere sulla stampante la copia della pagina video o di una particolare finestra;
- "la" che permette di scrivere una stringa delle dimensioni volute a partire da un punto qualsiasi dello schermo di cui si specifichino le coordinate;
- "pk" che permette di ridefinire il codice ASCII relativo ad un qualunque tasto della consolle (nel nostro caso è particolarmente utile a causa della mancanza di un ben preciso tasto di back space nella tastiera dell'M20);

Mediante l'invio di una opportuna stringa di caratteri vengono inoltre predisposti i parametri della stampante collegata (72 linee per modulo e salto carta alla fine del modulo stesso). Infine, per mezzo dell'istruzione Window, vengono aperte alcune finestre video che conterranno l'intestazione della mostra e, di volta in volta, i vari messaggi di aiuto per il pubblico: a tale proposito occorre ricordare che per semplicità d'uso, i tasti di return e di back space sono sempre indicati, rispettivamente, come "il tasto VERDE" e "il tasto ROSSO".
La seconda parte del programma si occupa invece dell'adescamento del pubblico e del collegamento tra i vari programmi.
In assenza di qualsiasi attenzione da parte dei visitatori, sul video compaiono alternativamente due messaggi di Oulipoit che invitano a fare la sua conoscenza; non appena però qualcuno si avvicina e preme il fatidico tasto VERDE, ecco che compare la lista di tutte le possibili scelte, e basta digitare un numero per indicare quella preferita.

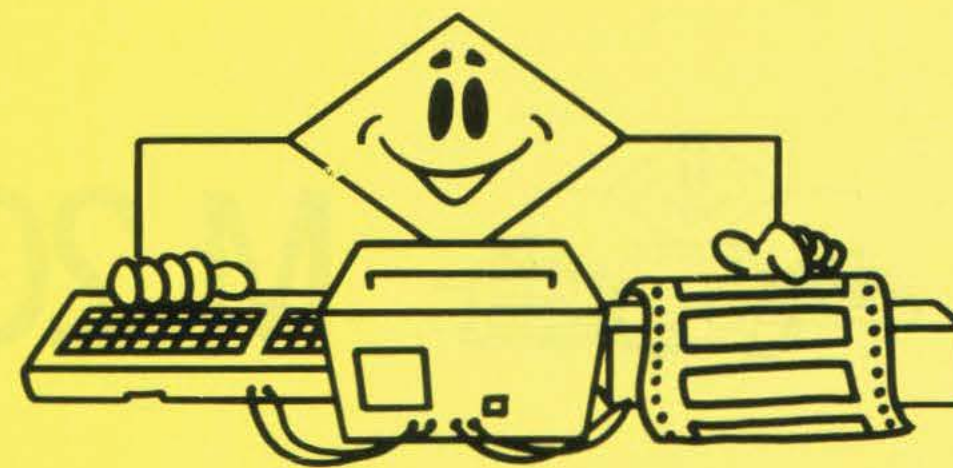
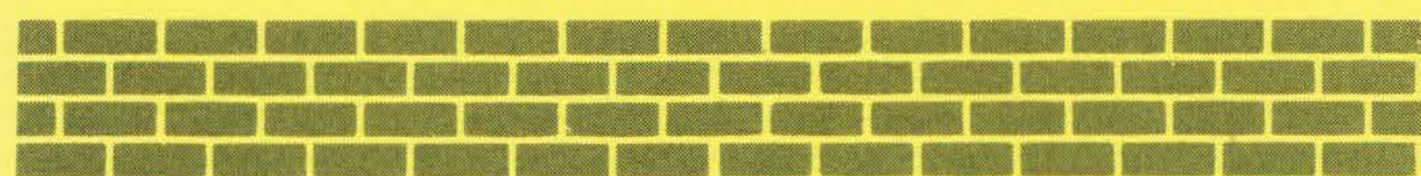
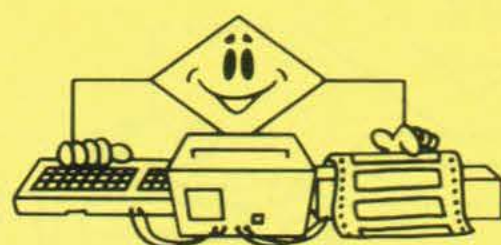



Figura 1 - Il listato del programma chiamante.

```
2 REM PILOTA
10 IF ANT%>0 THEN GOTO 70
20 PRINT:INPUT"che giorno e' oggi (in cifre) -->";GIT%
30 PRINT:INPUT"che mese (in cifre) -->";MET%
40 PRINT:INPUT"che anno -->";ANT%
50 IF ANT%<100 THEN ANT%=ANT%+1900
60 COMMON GIT%,MET%,ANT%
70 CLOSE
80 CLOSE WINDOW
90 EXEC"-dcons:"
100 DIM SCELTA$(5)
110 ON ERROR GOTO 770
120 DATA "OTTIMA SCELTA...","E VA BE'...",...
130 FOR I=1 TO 5
140 READ SCELTA$(I)
150 NEXT I
160 EXEC "pl sp"
170 EXEC "pl la"
180 EXEC "pl pk"
190 EXEC "sf,,72"
200 EXEC "+dcons:"
210 CALL"pk &5E,&08"
220 LPRINT CHR$(27);"Q072";CHR$(27);"Z";
230 CLS
240 I=WINDOW (0,54)
250 WINDOW %2
260 LINE (0,0)-(512,256),,B
270 EXEC "la 'JARRY E LA PATAFISICA',50,17,3,0"
280 EXEC"LA 'MILANO PALAZZO REALE MAG-AGO/83',54,2,2,0"
 290 WINDOW %1:CLOSE WINDOW %4:CLOSE WINDOW %3:CLS
300 REM CALL"sp 2"
310 CLS
320 R=1
330 ON R GOSUB 780,920
340 A$=""
350 WHILE I<9000 AND A$=""
360 I=I+1
370 A$=INKEY$
380 WEND
390 IF LEN(A$)=0 THEN GOTO 330
400 CLS
410 PRINT
 420 PRINT "    La tua fiducia non e' mal riposta: So fare molte cose"
430 PRINT "1- Generare anatemi o fare importanti rivelazioni sulla vita"
440 PRINT "2- Spaziare oltre la saggezza umana e generare proverbi inediti"
450 PRINT "3- Le stelle non sbagliano e io le leggo per voi: l'oroscopo"
460 PRINT "4- Dare nuovi suoni alla lettura di illustri opere"
470 PRINT "5- Reinterpretare testi letterari"
480 PRINT "6- Abbreviare le poesie troppo lunghe"
490 PRINT "7- Raccontare una storia..."
500 M=WINDOW (1,175)
510 S=WINDOW (1,155)
```

M 20

Seguito programma chiamante.

```
520 WINDOW %3
530 LINE (0,0)-(512,256),,B
540 EXEC"la'Premi il numero corrispondente alla scelta. Puoi correggere con il
   tasto ROSSO',5,13,1,0"
 550 CALL"la"("Ricordati dopo ogni risposta di premere il tasto VERDE",5,2,1,0)
560 WINDOW %4
570 CLS:PRINT "                                                    Cosa preferisci? -->";
580 GOSUB 1130
 590 IF T=9000 THEN GOTO 290 ELSE S$=BUF$
600 IF LEN(S$)>1 THEN PRINT CHR$(7):GOTO 570
610 IF S$<"1" OR S$>"7" THEN PRINT CHR$(7);:GOTO 570
620 CLS
630 CLS %3:CLOSE WINDOW %3:CLS %4:CLOSE WINDOW %4
640 WINDOW %1
650 RANDOMIZE(T)
660 C=RND*100 MOD 5+1
670 CLS:LINE (0,0)-(512,256),,B
680 CALL"la"(SCELTA$(C),20,100,2,0)
690 CALL"la"("aspetta solo un attimo...",20,50,2,0)
700 OPEN "r",£1,"pilota:stat",2 : FIELD £1,2 AS RICH$
710 GET £1,VAL(S$) : LSET RICH$=MKI$(CVI(RICH$)+1)
720 PUT £1,VAL(S$)
730 CLOSE £1
740 ON VAL(S$) GOTO 1050,1060,1070,1080,1090,1100,1110
750 CALL"pk &5E"
760 END
770 RESUME NEXT
780 CLS
790 PRINT
800 PRINT "     Buon giorno, umano."
810 PRINT "     Sono OULIPOIT, programma funzionante sul calcolatore M20"
 820 PRINT "     Ho l'obiettivo di soppiantare i vostri intellettuali,"
830 PRINT "     in quanto mi sostituiro' a loro nel comporre poesie,"
 840 PRINT "     nel raccontarvi storie, nel prevedere il vostro futuro"
 850 PRINT "     e fare tutto cio' che oggi un letterato cerca di fare."
 860 PRINT "     Sembro presuntuoso? No! So essere affabile e amichevole."
870 PRINT "     Anzi,facciamo conoscenza; siediti davanti a me"
880 PRINT "     e premi il tasto VERDE"
890 R=2
900 I=0
910 RETURN
920 CLS
930 PRINT
940 PRINT "  Nessuno vuole parlare con me!"
950 PRINT "  E pensare a quanto c'e' voluto per pensarmi e costruirmi!"
960 PRINT "  Ai sacrifici che hanno fatto i miei autori!"
970 PRINT "  E poi io so divertire!"
980 PRINT "  Come diceva il mio amico Bill,"
990 PRINT "   'la pazzia,mio Signore,regna su questa piazza,"
1000 PRINT "    e non v'e' luogo al mondo ov' essa non risplenda!'"
1010 PRINT "  Non abbandonatemi! Sedetevi e premete il tasto VERDE"
1020 R=1
1030 I=0
```

Seguito programma chiamante.

```
1040 RETURN
1050 CHAIN "pilota:AFORISMI
1060 CHAIN "pilota:PROVERBI
1070 CHAIN "pilota:STELLE
1080 CHAIN "pilota:SCAMBI
1090 CHAIN "pilota:MANIP
1100 CHAIN "pilota:ULTIME
1110 CHAIN "pilota:ROMANZI
1120 REM ROUTINE DI INPUT TEMPORIZZATO
1130 BUF$=""
1140 A$="":T=0
1150 WHILE T<9000 AND A$=""
1160 T=T+1
1170 A$=INKEY$
1180 WEND
1190 IF T=9000 THEN GOTO 1240
1200 IF (LEN(BUF$)>0) AND (ASC(A$)=8) THEN BUF$=LEFT$(BUF$,
    LEN(BUF$)-1):PRINT CHR$(8);" ";CHR$(8);:GOTO 1140
 1210 IF (ASC(A$)=13) THEN GOTO 1240
1220 IF ASC(A$)<>8  THEN BUF$=BUF$+A$ : PRINT A$;
 1230 IF LEN(BUF$)<=1 THEN GOTO 1140
1240 RETURN
```

In seguito ad una richiesta, viene caricato in memoria il programma corrispondente il quale, dopo l'esecuzione della dedica, ricaricherà in memoria di nuovo il programma PILOTA.
Tutto questo scambio di memoria tra un programma e l'altro è possibile grazie al comando BASIC CHAIN, che permette la concatenazione di più programmi e lo scambio, tra di loro, di un certo numero di variabili (tutte quelle definite come "COMMON" all'interno dei programmi stessi). Ad ogni selezione vengono inoltre incrementati i contatori posti sul file "stat" che conterrà, perciò, sempre il numero totale delle selezioni effettuate, mentre i tempi morti durante il caricamento dei vari programmi vengono riempiti con alcune frasi di commento generate casualmente.

Un ultimo importante particolare è rappresentato dalla routine di input temporizzato. Uno dei pericoli del dialogo col pubblico per mezzo dell'istruzione INPUT era costituito dal fatto che, se un visitatore, nel bel mezzo dell'introduzione di una stringa di risposta, avesse deciso che non era il caso di "perdere altro tempo con tali sciocchezze", Oulipoit sarebbe rimasto bloccato a quel punto fino a quando qualche buon'anima non si fosse premurata di battere il tasto di return. Per ovviare a questo inconveniente si è perciò dovuto costruire una routine di input con un time out di circa un minuto tra un carattere e l'altro: se entro tale tempo nessun tasto viene più battuto, tutta la situazione torna nelle mani del pilota

Tale routine, insieme con le istruzioni di apertura e chiusura di finestre su video e di stampa di stringhe particolari, è presente in ogni altro programma di Oulipoit e costituisce ciò che abbiamo definito come una ramificazione del PILOTA. Comunque la maggior parte di queste particolari soluzioni sono agevolmente sostituibili con altre equivalenti presenti in quasi tutti i personal.

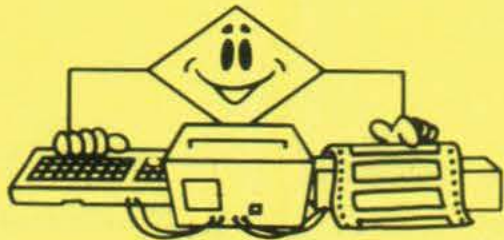
Aforismi

L'algoritmo su cui si fonda il programma AFORISMI è già stato ampiamente discusso nell'articolo di presentazione di Oulipoit apparso su **Bit** (n. 43), perciò ci limiteremo a riassumerlo brevemente.
Ogni linguaggio è basato su di un certo insieme di parole o di simboli (il vocabolario); alcune sequenze di tali parole sono riconosciute come corrette e costituiscono le frasi del linguaggio mentre altre, al contrario, sono giudicate scorrette oppure mal costruite. Ciò che permette di determinare se una particolare sequenza di parole costituisce o meno una frase è la sintassi del linguaggio stesso.

A partire da un certo insieme di regole sintattiche consistenti e da un vocabolario iniziale, è possibile, in base a ciò che si è detto, generare, per esempio in modo casuale, tutte (avendone la pazienza!) le frasi ammesse da un qualunque linguaggio. Questo è il compito del programma AFORISMI.
Il vocabolario di cui dispone il programma è tutto contenuto in un certo numero di file interni creati mediante l'istruzione "DATA", e le parole che vi appartengono, per semplificare al massimo la corretta applicazione delle norme della sintassi, sono divise per categorie grammaticali (articoli, nomi, avverbi, ecc.).
I termini che costituiscono il patrimonio lessicale di AFORISMI sono 338, numero più che sufficiente per creare un'infinità di differenti frasi, tuttavia ne sono riportati solo alcuni a titolo di esempio: ognuno si sbizzarrisca come può.
Anche la sintassi del nostro linguaggio è piuttosto ristretta; ogni frase può essere costruita in uno dei seguenti modi:

- ARTICOLO + NOME + VERBO TRANSITIVO + COMPLEMENTO OGGETTO + AVVERBIO
- ARTICOLO + NOME + VERBO INTRANSITIVO + AVVERBIO

inoltre la parte nominale di ogni frase (articolo + nome) può essere anche costituita da una particolare esortazione del tipo "sia maledetto colui che" oppure "lunga vita a", e all'inizio di ogni proposizione affermativa è possibile la presenza di un complemento di luogo.

Dopo una parte iniziale, in cui tutti i termini del vocabolario vengono caricati in 12 array (uno per ogni classe grammaticale rappresentata) per permettere una più agevole ricerca casuale,

il programma segue uno sviluppo ad albero binario.
Come prima cosa esso opera una scelta tra la costruzione di una frase affermativa oppure, diciamo così, di un anatema (entrambi i costrutti hanno uguali probabilità di vedere la luce) e, una volta sciolto questo dilemma, viene generata di conseguenza la prima parte della frase.
La decisione successiva consiste nel continuare con un verbo transitivo oppure con uno intransitivo (anche in questo caso le probabilità sono salomonicamente uguali) e, se il fato propende per quest'ultimo, viene naturalmente omesso il complemento oggetto e si passa direttamente all'avverbio che, in molti casi, consiste in una locuzione avverbiale o in un complemento di modo.

La frase, ormai completa, rimane visualizzata sullo schermo per il tempo strettamente necessario per poterla meditare attentamente, quindi lascia il posto alla successiva: il programma, infatti, genera un gruppo di dieci frasi ogni volta che viene risvegliato e, ognuna di esse viene presentata al centro del video su righe di 50 caratteri al massimo (la variabile LAC contiene in ogni momento la lunghezza della stringa da visualizzare).
Anche in "AFORISMI", come in ogni altro programma di Oulipoit, sono presenti tutte le istruzioni di interfaccia col visitatore: il comando "la" per la stampa di stringhe di diversi formati, il comando "sp" per la copia sulla stampante del contenuto di una finestra video, l'istruzione WINDOW ecc. È naturalmente presente anche la routine di input temporizzato indispensabile per la richiesta del nome da inserire nella dedica.
Un'ultima annotazione: tutte le scelte aleatorie che il programma implica sono effettuate, come è facile immaginare, utilizzando l'arcinota funzione RND e il generatore di numeri casuali è inizializzato una volta per tutte con il prodotto tra i minuti primi e secondi forniti dalla procedura TIME$ al momento del richiamo del programma stesso.

Romanzi

Gli ingredienti necessari e sufficienti per costruire un romanzo d'appendice di discreto successo non sono molti e, fortunatamente, sono anche ben classificabili:

- i buoni;
- i cattivi;
- l'amore a prima vista;
- l'odio fino all'ultimo respiro.

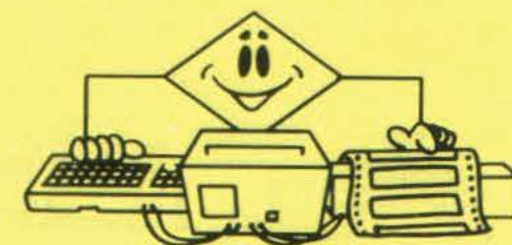
Il segreto sta poi nel saperli mescolare vigorosamente fino ad ottenere, tra un colpo di scena e l'altro, il più piacevole dei lieto fine. Tutti ormai si saranno accorti che, se si tratta di ingarbugliare le cose ad arte, non c'è nulla di meglio di un calcolatore, per cui esistono tutte le premesse perchè il programma "ROMANZI" possa sfornare trame romanzesche da manuale.
Il tutto ha origine, naturalmente, da un certo numero di personaggi (massimo sei e minimo due) che l'aspirante autore può qualificare mediante due attributi: il nome, che essendo una stringa di dimensioni qualsiasi, può contenere anche aggettivi, epiteti vari, vezzeggiativi e altre coloriture del genere, e il sesso a cui il personaggio appartiene. Ognuno di questi è inoltre descritto da alcune variabili che permettono di definirne lo stato durante ciascun episodio della trama, e che si evolvono con il naturale dipanarsi della vicenda.
Vediamo in dettaglio quale è il loro significato.
Lo stato di ogni personaggio è definito da:

- l'insieme dei CONOSCENTI: coloro cioè che, in un modo o nell'altro, sono entrati nella vita del nostro eroe
- l'insieme dei CONOSCIBILI: coloro che hanno buone probabilità di entrare, prima o poi, a far parte dei CONOSCENTI
- l'insieme degli SPOSABILI: tutti gli esponenti dell'altro sesso che non sono già coniugati
- l'insieme delle VITTIME: tutti gli appartenenti allo stesso sesso del personaggio e che egli, per ragioni di gelosia o di denaro, vorrebbe veder morti
- il CONIUGE, che indica se il personaggio può o meno far parte dell'insieme degli sposabili relativo a qualcun altro

L'evoluzione di queste variabili deve seguire, naturalmente, alcune regole di verosimiglianza e di convivenza civile e quindi, durante la vita del nostro eroe, vengono continuamente aggiornate per evitare inverosimili resurrezioni o imbarazzanti casi di poligamia.
Il nostro personaggio può dunque compiere una di queste tre azioni:

- può conoscere un appartenente all'insieme dei CONOSCIBILI,
- può impalmare un esponente degli SPOSABILI, eventualmente rompendo un precedente legame,
- può disfarsi di un pericoloso rivale oppure, ahimè, porre fine alla propria esistenza.

La storia si conclude con il tradizionale "...e vissero felici e contenti" quando lui e lei, superati tutti i colpi del destino, si ritrovano finalmente privi di nemici.
La prima parte del programma è dedicata all'inizializzazione delle variabili di stato e dell'introduzione degli attributi dei vari personaggi: due apposite routine si occupano di effettuare controlli sul nome e sul sesso a cui appartiene ogni singolo personaggio per evitare omonimie oppure la mancanza totale di esponenti di uno dei due sessi.

Il cuore del generatore di trame è invece costituito da un loop principale che, una volta costruiti l'array contenente tutti i personaggi ancora in vita, valuta le possibilità di azione di uno dei personaggi, scelto a caso, e richiama la routine relativa all'azione da compiere. Queste tre routine, corrispondenti a CONOSCERE, SPOSARE e UCCIDERE, si occupano a loro volta di aggiornare le variabili di stato dei personaggi coinvolti e di informare il pubblico, con frasi lapidarie, dell'azione avvenuta: ognuna di tali frasi è inoltre colorita con una affermazione casuale (scelta tra le 36 possibili) che ha il compito di chiarire le ragioni dell'accaduto.
Il problema principale incontrato nella stesura del programma "ROMANZI" è stato quello di trovare una valida rappresentazione per le variabili di stato che, intuitivamente, possono essere interpretate come insiemi, e che quindi devono essere trattate con le operazioni tipiche dell'insiemistica: unioni, intersezioni ecc. In assenza di una struttura di dati adeguata, tutte queste variabili

sono invece definite come interi: il BASIC implementato su M20 rappresenta infatti gli interi su due byte, e si hanno a disposizione ben 16 bit, ognuno dei quali rappresenta l'appartenenza o meno di un certo personaggio a quell'insieme. Vediamo di chiarire le cose con un esempio: se la variabile VITTIME del personaggio "tizio" vale 13 (rappresentazione binaria su due byte 0000000000001101), ciò significa che "tizio" non vede l'ora di sbarazzarsi del personaggio n. 1, del n. 2 e del n. 4.

Usando questo trucco, le operazioni di unione, di intersezione e di complemento possono essere facilmente implementate usando rispettivamente gli operatori AND, OR e NOT logici. In questo modo, rifacendoci alla definizione delle variabili di stato, definiremo le possibili vittime di "tizio" come:

VITTIME% = STESSOSEX%
AND CONOSCENTI%

e l'ennesimo bit del risultato si porterà al valore 1 solo se entrambi gli ennesimi bit dei due termini si trovano a 1.

Questo programma è passibile di ulteriori e interessanti modifiche e migliorie; innanzitutto è possibile farne di tutti i colori (di trame): rosa, gialle, nere, a luce rossa, semplicemente cambiando il nome alle azioni, modificando alcune frasi di coloritura e, eventualmente, qualche variabile di stato. Sarebbe inoltre interessante spezzare ogni azione in un certo numero di sotto azioni, in modo da ottenere un romanzo costituito da un certo numero di sotto romanzi, i quali a loro volta potrebbero essere costituiti da altrettanti sotto romanzi, i quali a loro volta ...

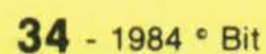
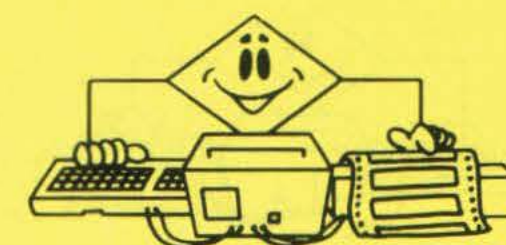
(Continua)

```
10 REM AFORISMI
20 COMMON NOME$,ANT%,MET%,GIT%
 30 CLS
40 M=WINDOW(1,175)
50 PRINT
60 PRINT:PRINT"Questa funzione aenera due tioi di frasi:"
70 PRINT
80 PRINT " Anatemi,maledizioni,benedizioni e simili"
90 PRINT:PRINT" Osservazioni,fatti,aneddoti,'oensierini'"
100 PRINT "    sulla vita umana"
110 PRINT:PRINT"            Dimmi come ti chiami ed io"
120 PRINT"            ti dedicherò ouesto mio oarto -->":
130 WINDOW %3
140 LINE (0,0)-(512,256),,B
150 EXEC"la'Se sbagli ouoi correggere con il tasto
    ROSSO',5,12,1,0"
160 EXEC "LA 'Ricordati dooo la risoosta di premere
        il tasto VERDE',5,2,1,0"
170 WINDOW %1
180 GOSUB 2310
190 IF T=9000 THEN NOME$="":GOTO 2300 ELSE NOME$=BUF$
200 IF LEN(NOME$)=0 THEN NOME$="UNO SCONOSCIUTO"
210 CLS
220 PRINT : LPRINT
230 A=VAL(MID$(TIME$,4,2))
240 D=VAL(MID$(TIME$,7,2))
250 CLS %3
260 FIN=0
270 C=D*A
280 RANDOMIZE(C)
290 DIM B$(5),C$(5),M$(50),F$(50),T$(50),I$(50),G$(50),
       P$(50),A$(50),L$(20),Z$(20),AM$(20)
300 REM TESTATA E START
310 GOSUB 820
```

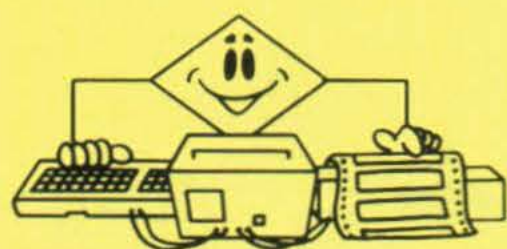
Figura 2 - L'insieme di istruzioni di aforismi.

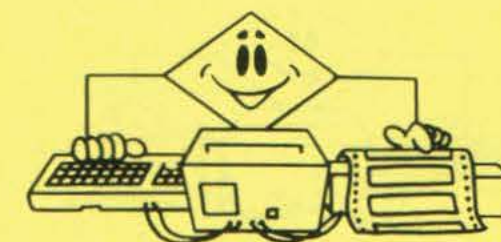
M 20

Seguito listato programma istruzione di aforismi.

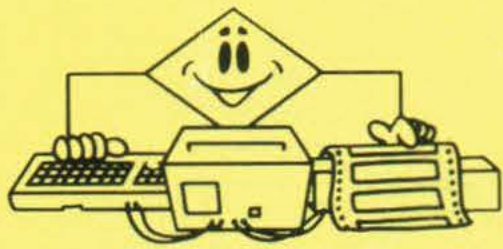
```
320 GOSUB 440
330 REM SELEZIONE TIPO FRASE
340 FOR RIT=0 TO 3000:NEXT RIT
350 CLS
360 LAC=0:PRINT:PRINT:PRINT:PRINT:PRINT:PRINT:LPRINT
370 IF RND>.3 THEN 410
380 REM GENERA FRASE NORMALE
390 GOSUB 890
400 IF FIN>10 THEN 2290 ELSE FIN=FIN+1: GOTO 330
410 REM GENERA ANATEMA
420 GOSUB 1690
430 IF FIN>10 THEN 2290 ELSE FIN=FIN+1 :GOTO 330
440 REM GENERA TESTATE E LETTURE INIZIALI *********
450 REM ARTICOLI MASCHILI
460 READ B1
470 FOR I=1 TO B1 : READ B$(I) : NEXT I
480 REM ARTICOLI FEMMINILI
490 READ C1
500 FOR I=1 TO C1 : READ C$(I) : NEXT I
510 REM NOMI MASCHILI
520 READ M1
530 FOR I=1 TO M1 : READ M$(I) : NEXT I
540  REM NOMI FEMMINILI
550 READ F1
560 FOR I=1 TO F1 : READ F$(I) : NEXT I
570 REM VERBI TRANSITIVI
580 READ T1
590 FOR I=1 TO T1 : READ T$(I) : NEXT I
600 REM VERBI INTRANSITIVI
610 READ I1
620 FOR I=1 TO I1 : READ I$(I) : NEXT I
630 REM AVVERBI
640 READ A1
650 FOR I=1 TO A1 : READ A$(I) : NEXT I
660 REM ANATEMA INIZIALE
670 READ G1
680 FOR I=1 TO G1 : READ G$(I) : NEXT I
690 REM CENTRO ANATEMA
700 READ P1
710 FOR I=1 TO P1 : READ P$(I) : NEXT I
720 REM LUOGO
730 READ L1
740 FOR I=1 TO L1 : READ L$(I) : NEXT I
750 REM AGGETTIVI FEMMINILI
760 READ Z1
770 FOR I=1 TO Z1 : READ Z$(I) : NEXT I
780 REM AGGETTIVI MASCHILI
790 READ AM1
800 FOR I=1 TO AM1 : READ AM$(I) : NEXT I
810 RETURN
820 REM FINE LETTURE
830 CLS
```

Seguito listato programma istruzione di aforismi.

```
840  CALL "LA"("SILENZIO' COMPONGO",20,30,4)
850 REM EXEC "SF 2
860 PRINT:PRINT:LPRINT
870 RETURN
880 PRINT
890 REM GENERA FRASE NORMALE ***********
900 IF RND(1)>.5 THEN 920
910 GOSUB 1640
920 REM GENERA SOGGETTO
930 GOSUB 1050
940 REM GENERA VERBO
950 IF RND(1)>.5 THEN 990
960 REM INTRANSITIVO
970 GOSUB 1190
980 GOTO 1020
990 REM TRANSITIVO
1000 GOSUB 1240
1010 GOSUB 1050
1020 REM AVVERBIO
1030 GOSUB 1290
1040 RETURN
1050 REM GENERA PARTE NOMINALE *************
1060 IF RND(1)>.5 THEN 1130
1070 REM MASCHILE-ARTICOLO-NOME
1080 GOSUB 1340
1090 GOSUB 1440
1100 IF RND(1)>.5 THEN 1180
1110 GOSUB 1390
1120 GOTO 1180
1130 REM FEMMINILE-ARTICOLO-NOME
1140 GOSUB 1490
1150 GOSUB 1590
1160 IF RND(1)>.5 THEN 1180
1170 GOSUB 1540
1180 RETURN
1190 REM GENERA VERBO INTRANSITIVO ***********
1200 I=RND(1)*I1+.5
1210 LAC=LAC+LEN(I$(I))
1220 IF LAC>50 THEN PRINT:PRINT I$(I);" ";:LPRINT I$(I);
     " ";:LAC=LEN(I$(I)) ELSE PRINT I$(I);" ";:LPRINT I$(I);" ";
1230 RETURN
1240 REM GENERA VERBO TRANSITIVO ***************
1250 I=RND(1)*T1+.5
1260 LAC=LAC+LEN(T$(I))
1270 IF LAC>50 THEN PRINT:PRINT T$(I);" ";:LPRINT T$(I);
     " ";:LAC=LEN(T$(I)) ELSE PRINT T$(I);" ";:LPRINT T$(I);" ";
1280 RETURN
1290 REM GENERA AVVERBIO *****************
1300 I=RND(1)*A1+.5
1310 LAC=LAC+LEN(A$(I))
1320 IF LAC>50 THEN PRINT:PRINT A$(I):LPRINT A$(I):
     LAC=LEN(A$(I)) ELSE PRINT A$(I):LPRINT A$(I)
```

M 20

Seguito listato programma istruzione di aforismi.

```
1330 RETURN
1340 REM GENERA ARTICOLO MASCHILE *********
1350 I=RND(1)*B1+.5
1360 LAC=LAC+LEN(B$(I))
 1370 IF LAC>50 THEN PRINT:PRINT B$(I):" ": LPRINT B$(I):
     " "::LAC=LEN(B$(I)) ELSE PRINT B$(I):" ": LPRINT B$(I):" ".
1380 RETURN
1390 REM GENERA AGGETTIVO MASCHILE ************
1400 IF CINT(I)<3 THEN I=RND(1)*11+.5 ELSE I=RND(1)*2+11.5
1410 LAC=LAC+LEN(AM$(I))
1420  IF LAC>50 THEN PRINT:PRINT AM$(I):" "::LPRINT AM$(I):
      " "::LAC=LEN(AM$(I)) ELSE PRINT AM$(I):" "::LPRINT AM$(I):" ":
1430 RETURN
1440 REM GENERA NOME MASCHILE *********
1450 IF CINT(I)<3 THEN  I=RND(1)*19+.5 ELSE I=RND(1)*11+19.5
1460 LAC=LAC+LEN(M$(I))
1470 IF LAC>50 THEN PRINT:PRINT M$(I):" ": LPRINT M$(I):
     " "::LAC=LEN(M$(I)) ELSE PRINT M$(I):" "::LPRINT M$(I):" ":
1480 RETURN
1490 REM GENERA ARTICOLO FEMMINILE ************
1500 I=RND(1)*C1+.5
1510 LAC=LAC+LEN(C$(I))
1520 IF LAC>50 THEN PRINT:PRINT C$(I):" "::LPRINT C$(I):
     " "::LAC=LEN(C$(I)) ELSE PRINT C$(I):" ":.LPRINT C$(I):" ":
1530 RETURN
1540 REM GENERA AGGETTIVO FEMMINILE ****************
1550 I=RND(1)*Z1+.5
1560 LAC=LAC+LEN(Z$(I))
1570 IF LAC>50 THEN PRINT:PRINT Z$(I):" ":.LPRINT Z$(I):
      " ":.LAC=LEN(Z$(I)) ELSE PRINT Z$(I):" "::LPRINT Z$(I):" ":
1580 RETURN
1590 REM GENERA NOME FEMMINILE ***********
1600 I=RND(1)*F1+.5
1610 LAC=LAC+LEN(F$(I))
1620 IF LAC>50 THEN PRINT:PRINT F$(I):" ":.LPRINT F$(I):
     " "::LAC=LEN(F$(I)) ELSE PRINT F$(I):" "::LPRINT F$(I):" ":
1630 RETURN
1640 REM GENERA COMPLEMENTO DI LUOGO ***************
1650 I=RND(1)*L1+.5
1660 LAC=LAC+LEN(L$(I))
1670 IF LAC>50 THEN PRINT:PRINT L$(I):" "::LPRINT L$(I):
     " "::  LAC=LEN(L$(I)) ELSE PRINT L$(I):" ":.LPRINT L$(I):" ":
1680 RETURN
1690 REM GENERA ANATEMA **************
1700 REM PARTE INIZIALE
1710 GOSUB 1840
1720 REM PARTE CENTRALE
1730 GOSUB 1890
1740 IF RND(1)>.5 THEN 1780
1750 REM INTRANSITIVO
1760 GOSUB 1190
```

Seguito listato programma istruzione di aforismi.

```
1770 GOTO 1810
1780 REM TRANSITIVO
1790 GOSUB 1240
1800 GOSUB 1050
1810 REM AVVERBIO
1820 GOSUB 1290
1830 RETURN
1840 REM GENERA ANATEMA INIZIALE *************
1850 I=RND(1)*G1+.5
1860 LAC=LAC+LEN(G$(I))
1870 IF LAC>50 THEN PRINT:PRINT G$(I);" ";:LPRINT G$(I);
     " ";:   LAC=LEN(G$(I)) ELSE PRINT G$(I);" ";:LPRINT G$(I);" ";
1880 RETURN
1890 REM ANATEMA CENTRALE **************
1900 I=RND(1)*P1+.5
1910 LAC=LAC+LEN(P$(I))
1920 IF LAC>50 THEN PRINT:PRINT P$(I);" ";:LPRINT P$(I);
     " ";:LAC=LEN(P$(I)) ELSE PRINT P$(I);" ";:LPRINT P$(I);" ";
1930 RETURN
1940 REM DATI ****************
1950 DATA 4 : REM ARTICOLI MASCHILI
1960 DATA "IL","UN","LO","UNO"
1970 DATA 2 : REM ARTICOLI FEMMINILI
1980 DATA "LA","UNA"
1990 DATA 30 : REM NOMI MASCHILI
2000 DATA "CANE","GATTO","POMO","MANISCALCO","CAVALLO"
2010 DATA "TAFIRO",.......
2020 DATA 41 : REM NOMI FEMMINILI
2030 DATA "GATTA","FOCA","PADELLA","MENSOLA","SCOPA"
2040 DATA "SINEDDOCHE",.......
2050 DATA 40 : REM VERBI TRANSITIVI
2060 DATA "SCUOTE","ISTIGA","PREVARICA","CONSOLA"
2070 DATA "MANDA IN SOLLUCCHERO",.......
2080 DATA 47 : REM VERBI INTRANSITIVI
2090 DATA "SI CROGIOLA","SI SPRECA","FA FLANELLA"
2100 DATA "AGONIZZA",.......
2110 DATA 49 : REM AVVERBI
2120 DATA "CON UN GRIDO","PER LA PATRIA","SPONTANEAMENTE"
2130 DATA "SENZA PAURA",.......
2140 DATA 43 : REM ANATEMI INIZIALI
2150 DATA "LODE E GLORIA A","VIVRA'IN ETERNO"
2160 DATA "NON TEME LA MORTE",.......
2170 DATA 42 : REM ANATEMI CENTRALI
2180 DATA "CHI","COLUI CHE","COLUI IL QUALE"
2190 DATA "IL POVERO CHE",.......
2200 DATA 12 : REM COMPLEMENTI DI LUOGO
2210 DATA "PER LA STRADA","SUL TRAM","IN CHIESA"
2220 DATA "IN AUTO",.......
2230 DATA 15 : REM AGGETTIVI FEMMINILI
2240 DATA "SELVAGGIA","DANNATA","SVAMPITA","MERAVIGLIOSA"
2250 DATA "SCHIZZINOSA",.......
```

M 20

Seguito listato programma istruzione di aforismi.

```
2260 DATA 13 : REM AGGETTIVI MASCHILI
2270 DATA "BELLICOSO","VIRILE","PELOSO","CICCIONE"
2280 DATA "POTENTE",.........
2290 CHAIN "pilota dedica"
2300 CHAIN "pilota.init.bas"
2305 REM ROUTINE DI INPUT TEMPORIZZATO
2310 BUF$=""
2320 A$="":T=0
2330 WHILE T<9000 AND A$=""
2340 T=T+1
2350 A$=INKEY$
2360 WEND
2370 IF T=9000 THEN GOTO 2420
2380 IF (LEN(BUF$)<>0) AND (ASC(A$)=8) THEN BUF$=LEFT$(BUF$
     ,LEN(BUF$)-1):PRINT CHR$(8);" ";CHR$(8);:GOTO 2320
2390 IF (ASC(A$)=13) THEN GOTO 2420
2400 IF ASC(A$)<>8 THEN BUF$=BUF$+A$:PRINT A$;
2410 GOTO 2320
2420 RETURN
```

Figura 3 - Il listato che produce interessanti romanzi.

```
10 REM ROMANZI
20 COMMON ANT%,MET%,GIT%,NOME$
30 CLS
40 OPTION BASE 1
50 DIM NOME$(8),SESSO$(8),CONIUGE%(8),CONOSCENTI%(8),
   CHANCE$(3),ARR%(8)
60 DIM COMM$(36)
70 COMM$(1)="ISTINTIVAMENTE "
80 COMM$(2)="SOTTO L'IMPULSO DELLA GELOSIA
90 COMM$(3)="IN UN MOMENTO DI SCONFORTO "
100 COMM$(4)="...
110 COMM$(20)="SFIDANDO LA MORTE "
120 COMM$(26)="PER UNA BEFFA DEL DESTINO "
130 COMM$(33)="IN UNA NOTTE BUIA E TEMPESTOSA "
140 SEED=LEN(TITOLO$)
150 M=WINDOW(1,175)
160 PRINT
170 PRINT "finalmente un po' di spazio alla mia creativita' "
180 PRINT "Vi raccontero' una storia "
190 PRINT "Io posso intrecciare i destini dei personaggi che"
200 PRINT "voi mi indicherete."
210 PRINT "Amici,nemici,personaggi famosi,popoli,cose"
220 PRINT "agiranno nelle mie trame."
230 WINDOW %3
240 LINE(0,0)-(512,256),,B
250 CALL"la"("Ricordati dopo ogni risposta di premere il
    tasto VERDE",5,2,1,0)
260 CALL"LA"("Puoi correggere con il tasto ROSSO",5,12,1,0)
270 WINDOW %1
280 PRINT:PRINT"Dimmi come ti chiami ed io"
290 PRINT"ti dedichero' questo mio parto -->";
```

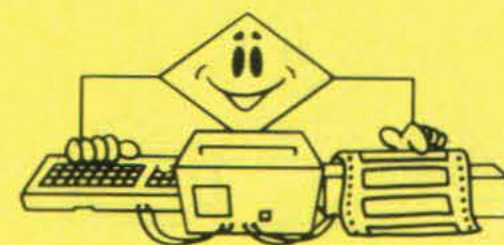
```
300 GOSUB 2130
310 IF T=9000 THEN NOME$="":GOTO 2100 ELSE NOME$=BUF$
320 IF LEN(NOME$)=0 THEN NOME$="UNO SCONOSCIUTO"
330 CLS:WINDOW %3
340 EXEC"la'ho delle domande da farti',10,30,2,0"
350 FOR I=1 TO 2000:NEXT I
360 CLS
370 WINDOW %3
380 LINE(0,0)-(512,256),,B
390 EXEC"la'Se sbagli puoi correggere con il tasto
    ROSSO',5,12,1,0"
400 EXEC"la'Dopo ogni risposta ricordati di premere il
    tasto VERDE',5,2,1,0"
410 WINDOW %1
420 PRINT
430 CLS:PRINT
440 PRINT"Che titolo vuoi dare al romanzo? --)";
450 GOSUB 2130
460 IF T=9000 THEN GOTO 2100 ELSE TITOLO$=BUF$
470 CLS:PRINT
480 MES$=""
490 PRINT MES$:PRINT"Quanti personaggi vuoi far agire?
                (max 6-min 2)              --)"
500 GOSUB 2130
510 IF T=9000 THEN GOTO 2100 ELSE NUMP$=BUF$
```

Seguito listato programma romanzi.

M 20

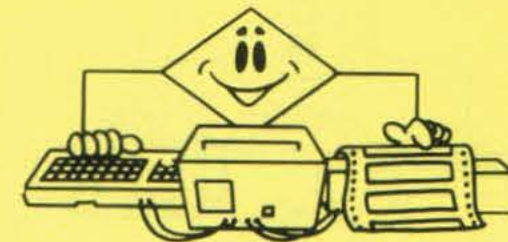
Seguito listato programma romanzi.

```
520 IF NUMP$<"2" THEN MES$="Non bastano ...devono esse e
    almeno due":PRINT:GOTO 490
530 IF NUMP$>"6" THEN MES$= "Sono troppi...Devono essere
    al massimo sei":PRINT:GOTO 490
540 CLS
550 UOMINI%=0
560 DONNE%=0
570 SPOSATI%=0
580 FOR TIZIO%=1 TO VAL(NUMP$)
590 PRINT:PRINT "Personaggio numero ";TIZIO%
600 CONIUGE(TIZIO%)=0
610 CONOSCENTI%(TIZIO%)=2^(TIZIO%-1)
620 IF TIZIO%<VAL(NUMP$) THEN GOTO 650
630 IF UOMINI%=0 THEN PRINT "deve essere un uomo!...
    PRINT:SESSO$(TIZIO%)="m":GOTO 660
640 IF DONNE%=0 THEN PRINT "deve essere una donna!... "
    PRINT:SESSO$(TIZIO%)="f":GOTO 660
650 GOSUB 1240
660 GOSUB 1310
670 IF (SESSO$(TIZIO%)=M$ OR SESSO$(TIZIO%)=M1$) THEN
    UOMINI%=UOMINI% OR 2^(TIZIO%-1) ELSE DONNE%=DONNE% OR 2^(TIZIO%-1)
680 CLS
690 SEED=SEED+LEN(NOME$(TIZIO%))
700 NEXT TIZIO%
710 CLS %3:CLOSE WINDOW %3
720 EXEC"la'Un attimo che stampo l'intestazione',10,50,2,0"
730 CALL"so 2"
740 CLS
750 RANDOMIZE(SEED)
760 CLS
770 LPRINT TITOLO$:PRINT TITOLO$
780 LPRINT:PRINT
790 LPRINT "PERSONAGGI ED INTERPRETI...":PRINT
    "PERSONAGGI ED INTERPRETI..."
800 FOR I=1 TO VAL(NUMP$)
810 LPRINT NOME$(I):PRINT NOME$(I)
820 NEXT I
830 LPRINT:PRINT:LPRINT:PRINT
840 FINITO=0
850 WHILE NOT FINITO
860 TUTTI%=DONNE% OR UOMINI%
870 IF TUTTI%=0 THEN FINITO=-1 :GOTO 1160
880 I%=0
890 J%=0
900 WHILE TUTTI%<>0
910 I%=I%+1
920 TEMP=TUTTI% MOD 2
930 IF TEMP=1 THEN J%=J%+1:ARR(J%)=I%
940 TUTTI%=TUTTI%\2
950 WEND
960 TIZIO%=ARR(((RND*100) MOD J%)+1)
970 IF (SESSO$(TIZIO%)="m" OR SESSO$(TIZIO%)="M") THEN
```

Seguito listato programma romanzi.

```
     STESSOSEX%=UOMINI% ALTROSEX%=DONNE% ELSE STESSOSEX%=DONNE% ALTROSEX%=UOMINI%
980 CONOSCENTI%(TIZIO%)=CONOSCENTI%(TIZIO%) AND (DONNE%
     OR UOMINI%)
990 SPOSABILI%=(ALTROSEX% AND CONOSCENTI%(TIZIO%)) AND
    (255-SPOSATI%)
1000 CONOSCIBILI%=(DONNE% OR UOMINI%) AND (255-CONOSCENTI%(TIZIO%))
1010 VITTIME%=STESSOSEX% AND CONOSCENTI%(TIZIO%) AND SPOSATI%
1020 CONG=(CONIUGE%(TIZIO%)=0) OR (CONOSCENTI%(TIZIO%)=2^(TIZIO%-1))
     OR (STESSOSEX%=2^(TIZIO%-1))
1030 IF CONG THEN VITTIME%=VITTIME% AND (255-2^(TIZIO%-1))
1040 IF CONIUGE%(TIZIO%)=0 THEN GOTO 1060
1050 IF ALTROSEX%<>2^(CONIUGE%(TIZIO%)-1) THEN VITTIME%=
     VITTIME% OR 2^(CONIUGE%(TIZIO%)-1)
1060 I%=0
1070 IF SPOSABILI%<>0 THEN I%=I%+1 CHANCE$(I%)="S"
1080 IF CONOSCIBILI%<>0 THEN I%=I%+1 CHANCE$(I%)="C"
1090 IF VITTIME%<>0 THEN I%=I%+1 CHANCE$(I%)="U"
1100 FINITO=I%=0
1110 IF FINITO THEN GOTO 1160
1120 CAS%=((RND*100) MOD I%)+1
```

M 20

Seguito listato programma romanzi.

```
1130 IF CHANCE$(CASO%)="C" THEN GOSUB 1490:GOTO 1160
1140 IF CHANCE$(CASO%)="S" THEN GOSUB 1660:GOTO 1160
1150 IF CHANCE$(CASO%)="U" THEN GOSUB 1890
1160 WEND
1170 LPRINT:LPRINT:PRINT:PRINT
1180 LPRINT "... ": PRINT "..."
1190 LPRINT "E "; NOME$(ARR(1));" E ";NOME$(ARR(2));
     " VISSERO FELICI E CONTENTI.":LPRINT:LPRINT:PRINT:PRINT
1200 PRINT "E ";NOME$(ARR(1));" E ";NOME$(ARR(2));
     "VISSERO FELICI E CONTENTI"
1210 CHAIN "dedica"
1220 CHAIN "pilota init.bas"
1230 REM LEGGE ED EFFETTUA CONTROLLI SUL SESSO DEL PERSONAGGIO
1240 PRINT : PRINT
1250 PRINT:PRINT "quale e' il sesso del personaggio?-->";
1260 GOSUB 2130
1270 IF T=9000  THEN GOTO 2100
1280 SESSO$(TIZIO%)=BUF$
1290 IF ((SESSO$(TIZIO%)<>"M") AND (SESSO$(TIZIO%)<>"m")) AND ((SESSO$(TIZIO%)<>"f")
    AND (SESSO$(TIZIO%)<>"F")) THEN PRINT CHR$(7):PRINT " prima specificare il sesso ":
 GOTO 1250 :1300 RETURN
1310 REM QUESTA ROUTINE LEGGE IL NOME DI OGNI PERSONAGGIO
     ED EFFETTUA CONTROLLI SULLA STRINGA DIGITATA
1320 PRINT
1330 OK=0
1340 WHILE NOT OK
1350 CONTRO$=""
1360 PRINT CONTRO$:PRINT "Come si chiama il personaggio? -->";
1370 GOSUB 2130
1380 IF T=9000 THEN GOTO 2100 ELSE NOME$(TIZIO%)=BUF$
1390 LUNG=LEN(NOME$(TIZIO%))
1400 OK=(LUNG>0) AND (SPACE$(LUNG)<>NOME$(TIZIO%))
1410 PRINT
1420 IF NOT OK THEN CONTRO$="almeno un carattere...!":
      PRINT:GOTO 1360
1430 FOR CAIO%=1 TO TIZIO%-1
1440 OK=OK AND (NOME$(TIZIO%)<>NOME$(CAIO%))
1450 NEXT CAIO%
1460 IF NOT OK THEN CONTRO$="nome gia' usato!...":GOTO 1360
1470 WEND
1480 RETURN
1490 REM QUESTA ROUTINE FA CONOSCERE A TIZIO UN NUOVO
     PERSONAGGIO
1500 I%=0
1510 J%=0
1520 WHILE CONOSCIBILI%<>0
1530 I%=I%+1
1540 TEMP=CONOSCIBILI% MOD 2
1550 IF TEMP=1 THEN J%=J%+1:ARR(J%)=I%
1560 CONOSCIBILI%=CONOSCIBILI%\2
1570 WEND
1580 CAIO%=ARR(((RND*100) MOD J%)+1)
```

```
1590 IF TIZIO%=CAIO% THEN GOTO 1560
1600 CONOSCENTI%(TIZIO%)=CONOSCENTI%(TIZIO%) OR 2^(CAIO%-1)
1610 CASO=(RND*100) MOD 35)+1
1620 LPRINT COMM$(CASO):PRINT COMM$(CASO);
1630 CONOSCENTI%(CAIO%)=CONOSCENTI%(CAIO%) OR 2^(TIZIO%-1)
1640 LPRINT NOME$(TIZIO%);" CONOSCE ";NOME$(CAIO%)1
     PRINT NOME$(TIZIO%);" CONOSCE ";NOME$(CAIO%)
1650 RETURN
1660 REM QUESTA ROUTINE FA SPOSARE TIZIO CON UN PERSONAGGIO
1670 I%=0
1680 J%=0
1690 WHILE SPOSABILI%<>0
1700 I%=I%+1
1710 TEMP=SPOSABILI% MOD 2
1720 IF TEMP=1 THEN J%=J%+1:ARR(J%)=I%
1730 SPOSABILI%=SPOSABILI%\2
1740 WEND
1750 CASO=((RND*100) MOD 36)+1
1760 CAIO%=ARR(((RND*100) MOD J%)+1)
1770 PRINT COMM$(CASO);NOME$(TIZIO%):LPRINT COMM$(CASO);
     NOME$(TIZIO%)
1780 IF CONIUGE%(TIZIO%)=0 THEN GOTO 1840
1790 LPRINT " LASCIA ";NOME$(CONIUGE%(TIZIO%));" PER ";
     NOME$(CAIO%)
```

Seguito listato programma romanzi.

M 20

Seguito listato programma romanzi.

```
1800 PRINT " LASCIA ";NOME$(CONIUGE%(TIZIO%));" PER ";
    NOME$(CAIO%)
1810 CONIUGE%(CONIUGE%(TIZIO%))=0
1820 SPOSATI%=SPOSATI% AND (255-2^(CONIUGE%(TIZIO%)-1))
1830 GOTO 1850
1840 LPRINT " SPOSA ";NOME$(CAIO%):PRINT " SPOSA ";NOME$(CAIO%)
1850 CONIUGE%(TIZIO%)=CAIO%
1860 CONIUGE%(CAIO%)=TIZIO%
1870 SPOSATI%=SPOSATI% OR (2^(CAIO%-1)) OR (2^(TIZIO%-1))
1880 RETURN
1890 REM QUESTA ROUTINE  FA UCCIDERE A TIZIO UN PERSONAGGIO
     (EVENTUALMENTE SE STESSO)
1900 I%=0
1910 J%=0
1920 WHILE VITTIME%<>0
1930 I%=I%+1
1940 TEMP=VITTIME% MOD 2
1950 IF TEMP=1 THEN J%=J%+1:ARR(J%)=I%
1960 VITTIME%=VITTIME%c2
1970 WEND
1980 CAIO%=ARR(((RND*100) MOD J%)+1)
1990 CASO=((RND*100) MOD 36)+1
2000 LPRINT COMM$(CASO);:PRINT COMM$(CASO);
2010 LPRINT NOME$(TIZIO%);:PRINT NOME$(TIZIO%);
2020 DONNE%=DONNE% AND (255-2^(CAIO%-1))
2030 UOMINI%=UOMINI% AND (255-2^(CAIO%-1))
2040 IF CONIUGE%(CAIO%)=0 THEN GOTO 2070
2050 CONIUGE%(CONIUGE%(CAIO%))=0
2060 SPOSATI%=SPOSATI% AND (255-2^(CAIO%-1)) AND
     (255-2^(CONIUGE%(CAIO%)-1))
 2070 IF TIZIO%=CAIO% THEN LPRINT " SI UCCIDE!! " ELSE LPRINT
     " UCCIDE ";NOME$(          CAIO%)
2080 IF TIZIO%=CAIO% THEN PRINT " SI UCCIDE!! " ELSE PRINT
     " UCCIDE ";NOME$(CAIO%)
2090 RETURN
2100 CHAIN "init.bas"
2110 RESUME NEXT
2120 REM ROUTINE DI INPUT TEMPORIZZATO
2130 BUF$=""
2140 A$="":T=0
2150 WHILE T<9000 AND A$=""
2160 T=T+1
2170 A$=INKEY$
2180 WEND
2190 IF T=9000 THEN GOTO 2240
2200 IF (LEN(BUF$)>0) AND (ASC(A$)=8) THEN BUF$=LEFT$(BUF$,
     LEN(BUF$)-1):PRINT CHR$(8);" ";CHR$(8);:GOTO 2140
2210 IF (ASC(A$)=13) THEN GOTO 2240
2220 IF ASC(A$)<>8 THEN BUF$=BUF$+A$:PRINT A$;
2230 GOTO 2140
2240 RETURN
```

METRO IMPORT

DIVISIONE INFORMATICA

Rivenditori Autorizzati:
SINCLAIR – COMMODORE – TEXAS – EPSON – SEIKOSHA – SAICO – JACKSON ED.

La **METRO IMPORT** nell'ambito della sua organizzazione, sempre all'avanguardia e in continua progressiva evoluzione sia qualitativa che tecnica, è in grado di fornire ai propri clienti, per corrispondenza o direttamente presso i punti vendita di Roma e Milano:

- **Una serie di home computers fra i più qualificati con i relativi accessori, software applicativi su cartridge, su nastro o su disco.**
- **Personal computers e periferiche con assistenza hardware da parte di personale specializzato.**
- **Assistenza software sia su pacchetti applicativi standard** (contabilità, fatturazione, magazzino, paghe e stipendi) **che per procedure personalizzate** (scientifiche e gestionali).
- **Leasing finanziario.**

Ogni realizzazione, dopo un accurato studio e sopralluogo, verrà consegnata "CHIAVI IN MANO":

METRO IMPORT 83
DIVISIONE INFORMATICA

Omaggio il catalogo di Informatica

Per ricevere il catalogo in omaggio, ritagliare e spedire il coupon allegando L. 500 in francobolli.

Ritagliare e spedire in busta chiusa a: METRO IMPORT s.n.c. - VIA DONATELLO, 37 - 00196 ROMA

Nome e Cognome ....

Indirizzo ....

C.A.P. .... Città ....

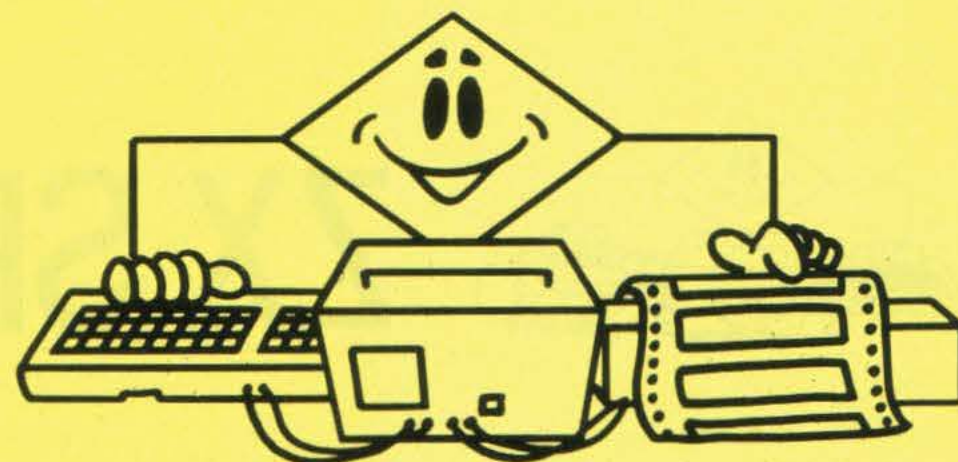
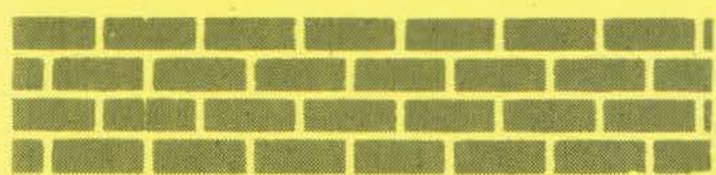
"!"

tecnologia americana

TELEVIDEO SISTEMS é importato e distribuito da:

MICROCOMP S.p.A.

00153 Roma · Viale M.Gelsomini, 28
tel. 06/57 78.484 - 57.78.324
telex: 616251

ZX SPECTRUM

ZIP: il gioco delle freccette

di **D. Citterio**

Provi il lettore ad immedesimarsi in questa situazione. Corre da me una sera la minore delle mie figlie, 7 anni e qualche giorno. "Papà, mi fai un gioco nuovo col computer?" Resto di sasso. "Come nuovo!? Te ne carico uno di quelli che abbiamo". "Ma no, io ne voglio uno che non ho mai fatto".

Mi sono sentito sull'orlo dell'abisso! Come padre e insegnante rispetto alla sua età rivesto i panni del "sotutto" di casa, colui al quale si ricorre in qualunque occasione; figuriamoci se potevo confessare di possedere doti di programmatore non proprio eccelse, e comunque non così fulminee da saper creare in pochi minuti un gioco. Il fatto è che a sette anni il padre è l'eroe per antonomasia e già mi preparavo a vedere questa mia figura franare con anni di anticipo. Quando balenò l'idea di Guglielmo Tell. Non mi si chieda da quale associazione di idee fu partorita, né da qual recondito anfratto mentale sia schizzata fuori. Probabilmente anche nei cieli già c'è un Santo che si occupa di informatica e dei suoi praticanti e a Lui credo di dovere il programma che vado ad illustrare.

È un giochino molto semplice. Consiste nel colpire un bersaglio posto su un lato dello schermo con una freccia che scocca dal lato opposto. È diviso in tre parti; nella prima il movimento della freccia è dal basso in alto, nella seconda da destra a sinistra, nella terza in diagonale. Il computer posiziona casualmente sia il bersaglio che l'arciere, al giocatore vien chiesto di effettuare l'allineamento muovendo l'arciere e quindi di far scoccare la freccia. Si hanno a disposizione cinque frecce per ogni parte ed alla fine vien fornito il punteggio.

Esso dipende dai centri realizzati e dal tempo trascorso; più tempo si impiega a far partire la freccia, meno punti si conquistano. Il gioco è stato pensato per bambini di otto/nove anni ed infatti la parte "scolastica" inserita effettua il controllo delle tabelline. Non potendo escludere che anche i più piccoli possano trarne motivo di impegno si può sostituire il tipo di calcolo proponendo alla linea 980 il segno di addizione invece di quello della moltiplicazione. Se cinque calcoli sembrano pochi si aumenti il 5 della linea 970 con l'avvertenza che se esso supererà il 10 bisognerà togliere uno dei due PRINT alla linea 980. Il gioco, molto lineare e all'apparenza fin troppo elementare, non deve trarre in inganno rispetto il suo reale livello di difficoltà. Al bambino vien chiesto di confrontare due posizioni che vanno poste in corrispondenza premendo i tasti N e M. Ciò presume una buona motricità fine della mano destra ed una parimenti buona coordinazione occhio-mano. La difficoltà aumenta nella seconda parte, quando gli stessi tasti controllano il movimento in verticale dell'arciere. Nell'ultima parte infine si noterà che non è affatto facile trovare per l'arciere la giusta posizione, ... e i lamenti del "trafitto" saranno numerosi. Inizialmente è opportuno raccomandare al bambino di controllare i tasti N e M con l'indice e il medio della mano destra, e di mantenere l'indice sinistro sul tasto Z per lanciare la freccia. In tempi successivi le dita possono essere variate per distribuire l'esercizio sull'intera mano. Ricordiamo che è sempre opportuno collegare lo Spectrum ad un piccolo amplificatore per apprezzare meglio il suono. Al termine di ogni parte il bambino è chiamato ad un breve intervallo di "lavoro scolastico".

Senz'altro sarà ben sopportato e potrebbe addirittura essere utile a stemperare quel minimo di tensione che sempre si ingenera durante un gioco.

REMarks

10-60	Presentazione del gioco.
70-100	Istruzioni.
120; 355; 540; 545	Scelta delle coordinate per posizionare bersaglio ed arciere.
170-230; 365-405; 560-600	Lettura della tastiera per il movimento dell'arciere.
535	Serve a scegliere se porre il bersaglio sull'asse delle ordinate o su quello delle ascisse.
820	DATA per creare l'albero ("a" grafica).
825	DATA per creare la mela ("b" grafica).
830-835	DATA per il personaggio che regge la mela ("c" e "d" grafiche).
840	DATA per creare la freccia orizzontale ("e" grafica).
845-850	DATA per creare l'arciere ("f" e "g" grafiche).
855	DATA per creare la freccia in diagonale ("h" grafica).
960-995	routine per gli esercizi di calcolo.

VARIABILI PRINCIPALI

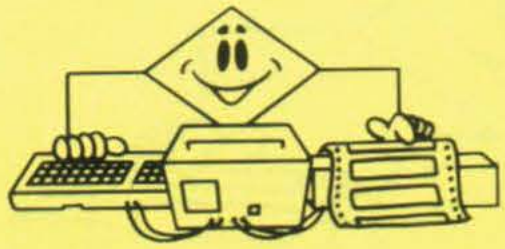
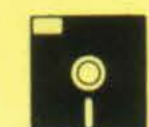
f	Contatore di frecce.
t	Tempo trascorso dalla stampa della freccia al suo scoccare.
x,y	Coordinate della freccia.
t1	Punti acquisiti ad ogni bersaglio colpito.
pt	Totalizzatore dei punti di ogni parte del gioco.
p	Penalità.
totale	Punteggio dell'intera partita.

ZX SPECTRUM

Figura 1 - Il listato del programma ZIP.

```
  10 PAPER 7: BORDER 5: CLS
  20 FOR n=0 TO 7: READ a: POKE
USR "a"+n,a: BEEP .05,n: NEXT n
  30 FOR x=8 TO 12: FOR y=0 TO 3
1: PRINT INK 4;AT x,y;CHR$ 144:
BEEP .01,y: NEXT y: NEXT x
  35 FOR m=1 TO 28 STEP 3
  40 FOR n=28 TO m STEP -1
  50 PRINT AT 10,n;"ZIP"; INK 4;
CHR$ 144
  60 BEEP m/100,n: NEXT n: NEXT
m
  70 CLS : PRINT TAB 11;"ISTRUZI
ONI": PRINT : PRINT "Il gioco co
nsiste nel colpire   una mela la
nciando delle frecce dopo  avert
e allineate con il   bersaglio."
  75 PRINT "Il gioco e' diviso i
n 3 parti.  Ogni volta ti daro'
5 frecce."
 100 PRINT "Bisogna lanciare in
fretta; piu'il tempo passa meno
punti si    guadagnano.": PRINT
"Il tasto N sposta la freccia a
 sinistra, il tasto M a destra.
 Il tasto Z la fa partire.": PRI
NT : PRINT "Ad ogni bersaglio ma
ncato perdi 5 punti.": PRINT AT
21,0;"Premi un tasto per incomin
ciare.": PAUSE 0
 102 PAPER 6: FOR n=0 TO 7: READ
 b: POKE USR "b"+n,b: BEEP .1,n:
 NEXT n
 105 LET p=0: LET pt=0: LET tota
le=0
 108 FOR f=1 TO 5
 110 LET t=0
 115 CLS : GO SUB 800
 120 LET a=INT (1+30*RND): LET y
=INT (1+30*RND)
 150 PRINT INK 1;AT 1,a;CHR$ 145
 160 LET t=t+1
 170 PRINT AT 19,y;" "
 180 IF INKEY$="" THEN GO TO 220
 190 IF INKEY$="n" THEN LET y=y-
1: IF y=0 THEN LET y=1
 200 IF INKEY$="m" THEN LET y=y+
1: IF y=31 THEN LET y=30
 210 IF INKEY$="z" THEN GO TO 24
0
 220 PRINT BRIGHT 1;AT 19,y;"↑"
 230 BEEP .03,0: PRINT AT 21,6;t
: GO TO 160
 240 FOR x=18 TO 1 STEP -1: PRIN
T AT x,y;"↑";AT x+1,y;" "
 250 BEEP .01,36: NEXT x
 260 IF a=y THEN GO SUB 900
 270 IF a<>y THEN GO SUB 910
 280 NEXT f
 285 PRINT AT 4,15;"prima parte"
: GO SUB 920
 290 GO SUB 960
 295 CLS : PRINT "Eccoti alla se
conda parte.","Ora la freccia ha
 movimento     orizzontale. In p
iu' vedi l'ar- ciere. Fai attenz
ione alla mela bersaglio; essa e
'  posta sulla  testa di un tuo a
mico. Se sbagliil colpo perdi 5
punti, ma se   trafiggi l'amico
ne perdi 10."
 298 PRINT AT 21,8;"Premi un tas
to": PAUSE 0
 300 BORDER 4: PAPER 6: CLS
 305 FOR n=0 TO 7: READ c: POKE
USR "c"+n,c: BEEP .05,n: NEXT n
 310 FOR n=0 TO 7: READ d: POKE
USR "d"+n,d: BEEP .05,n: NEXT n
 315 FOR n=0 TO 7: READ e: POKE
USR "e"+n,e: BEEP .05,n: NEXT n
 320 FOR n=0 TO 7: READ f: POKE
USR "f"+n,f: BEEP .05,n: NEXT n
 330 FOR n=0 TO 7: READ g: POKE
USR "g"+n,g: BEEP .05,n: NEXT n
 335 LET pt=0: LET p=0
 340 FOR f=1 TO 5
 345 LET t=0
 350 CLS : GO SUB 800
 355 LET a=INT (1+17*RND): LET x
=INT (1+18*RND)
 360 PRINT BRIGHT 1;AT a,1;CHR$
145; BRIGHT 0; INK 3;AT a+1,1;CH
R$ 146; INK 3;AT a+2,1;CHR$ 147
 365 LET t=t+1
 370 PRINT AT x,29;" ";AT x,30;"
";AT x+1,30;" "
 375 IF INKEY$="" THEN GO TO 395
 380 IF INKEY$="n" THEN LET x=x+
1: IF x=19 THEN LET x=18
 385 IF INKEY$="m" THEN LET x=x-
1: IF x=0 THEN LET x=1
 390 IF INKEY$="z" THEN GO TO 41
0
 395 PRINT AT x,30;CHR$ 149;AT x
+1,30;CHR$ 150
 400 PRINT BRIGHT 1;AT x,29;CHR$
148
 405 BEEP .03,0: PRINT AT 21,6;t
: GO TO 365
 410 FOR y=28 TO 1 STEP -1: PRIN
T AT x,y;CHR$ 148;" "
 415 BEEP .01,36: NEXT y
 420 IF a=x THEN GO SUB 900: NEX
T f: GO TO 450
 425 IF x=a+1 OR x=a+2 THEN GO S
UB 935: NEXT f: GO TO 450
 430 IF ATTR (x,y)=48 THEN GO SU
B 910: NEXT f
 450 PRINT AT 4,15;"seconda part
e": GO SUB 920
 455 GO SUB 960
 470 CLS : PRINT TAB 5;"Ultima p
arte del gioco"
 473 PRINT : PRINT : PRINT "Ora
la freccia scocca in diago- nale
. Colpire il bersaglio e'  prop
rio difficile.": PRINT : PRINT "
Premi un tasto per incominciare.
 475 PAUSE 0
 500 BORDER 2: CLS
 510 FOR n=0 TO 7: READ h: POKE
USR "h"+n,h: BEEP .05,n: NEXT n
 515 LET pt=0: LET p=0
 520 FOR f=1 TO 5
 525 LET t=0
 530 CLS : GO SUB 800
 535 LET w=INT (1+2*RND)
 540 IF w=1 THEN LET a=1: LET b=
INT (1+12*RND): GO TO 550
 545 IF w=2 THEN LET b=1: LET a=
INT (1+12*RND)
 550 PRINT BRIGHT 1;AT a,b;CHR$
145; BRIGHT 0; INK 3;AT a+1,b;CH
R$ 146;AT a+2,b;CHR$ 147
 555 LET x=18: LET y=INT (7+23*R
ND)
 560 LET t=t+1
 565 PRINT AT x,y;" ";AT x,y+1;"
";AT x+1,y+1;" "
 570 IF INKEY$="" THEN GO TO 590
 575 IF INKEY$="n" THEN LET y=y-
1: IF y=6 THEN LET y=7
 580 IF INKEY$="m" THEN LET y=y+
1: IF y=30 THEN LET y=29
 585 IF INKEY$="z" THEN GO TO 60
5
 590 PRINT BRIGHT 1;AT x,y;CHR$
151; BRIGHT 0;AT x,y+1;CHR$ 149;
AT x+1,y+1;CHR$ 150
 600 BEEP .02,0: PRINT AT 21,6;t
: GO TO 560
 605 FOR n=1 TO 17
 610 LET x=x-1: LET y=y-1
 612 IF ATTR (x,y)=51 THEN PRINT
AT x,y;CHR$ 151;AT x+1,y+1;" ":
GO SUB 935: NEXT f: GO TO 660
 615 PRINT AT x,y;CHR$ 151;AT x+
1,y+1;" ": BEEP .01,36
 620 IF x=1 OR y=1 THEN GO TO 63
0
 625 NEXT n
 630 IF a=x AND b=y THEN GO SUB
900: NEXT f: GO TO 660
 635 IF ATTR (x,y)=48 THEN GO SU
B 910
 650 NEXT f
 660 PRINT AT 4,15;"terza parte"
: GO SUB 920
 700 CLS
 710 PRINT : PRINT : PRINT "In q
uesta partita hai totalizza-to
PUNTI ";totale
 720 PRINT AT 15,3;"Vuoi giocare
 ancora? (S/N)"
 725 PAUSE 0
 730 IF INKEY$="s" THEN GO SUB 9
50: RUN
 740 STOP
 800 FOR n=0 TO 31: PRINT AT 0,n
;CHR$ 144;AT 20,n;CHR$ 144
 805 BEEP .01,n: NEXT n
 808 FOR n=1 TO 19: PRINT AT n,0
;CHR$ 144;AT n,31;CHR$ 144: BEEP
 .01,n: NEXT n
 810 PRINT AT 21,0;"Tempo      ▓P
enalita'   ▓TOT.     "
 815 RETURN
 820 DATA 126,235,182,219,173,90
,24,24
 825 DATA 0,32,16,8,12,22,30,12
 830 DATA 30,26,30,12,12,30,22,2
0
 835 DATA 18,30,12,12,12,15,31,0
 840 DATA 0,0,0,32,64,255,64,32
 845 DATA 15,39,39,71,66,191,71,
71
 850 DATA 39,39,3,3,3,15,15,0
 855 DATA 60,48,40,36,2,1,0,0
 900 BEEP .1,-24: PRINT AT 8,6;
FLASH 1;"BERSAGLIO COLPITO": LET
 t1=ABS (a-y)+INT (500/SQR t): L
ET pt=pt+t1
```

Seguito listato ZIP.

```
 905 PRINT AT 21,28;t1
 906 GO TO 945
 910 PRINT AT 6,6;"BERSAGLIO MAN
CATO": LET p=p+5: PRINT AT 21,20
;p
 915 GO TO 945
 920 PRINT AT 4,4;"Fine della ";
AT 8,4;"Hai totalizzato punti ";
pt;AT 9,6;"Le penalita' sono ";p
 925 PRINT AT 13,6;"Punteggio fi
nale ";pt-p
 928 LET totale=totale+pt-p
 930 GO TO 945
 935 PRINT AT 8,4;"Ahhhh...mi ha
i trafitto!"
 940 LET p=p+10: PRINT AT 21,20;
p
 945 PRINT AT 18,8;"Premi un tas
to"
 948 IF INKEY$="" THEN GO TO 948
 950 RETURN
 960 CLS
 965 PRINT TAB 5;"INTERVALLO SCO
LASTICO"
 970 FOR d=1 TO 5
 975 LET a=INT (1+10*RND): LET b
=INT (1+10*RND)
 980 PRINT : PRINT : PRINT ;"Qua
nto fa ";a;"x";b;"?";
 983 INPUT "Risultato ";c
 985 IF c=a*b THEN PRINT "    =";
c;"    esatto": NEXT d
 990 IF c<>a*b THEN GO TO 983
 995 PAUSE 100: RETURN
9000 CLS : PRINT FLASH 1;AT 10,0
;"         FERMA IL NASTRO
  "
9010 PRINT AT 21,0;"Premi un tas
to per incominciare.": PAUSE 0
9020 RUN
```

VIC 20

Poker: giochiamo d'azzardo!

Il gioco del Poker come qualsiasi altro gioco di carte può essere ben simulato dal computer. Per mezzo della funzione RND(x) si possono infatti creare le condizioni di casualità necessarie sia per la distribuzione delle carte che per l'imprevedibilità del gioco del computer stesso.

di **E. Morelli**

Questo programma di Poker per il VIC 20 fa giocare il computer, entro certi limiti, come un attento avversario.
Sono previste, in ordine di valore crescente, le seguenti combinazioni: minima, coppia, doppia coppia, tris, scala semplice (al re e all'asso), full, colore, poker e scala reale (al re e all'asso). Il capitale iniziale per il giocatore e per il computer è di un milione mentre la massima puntata singola è di 100.000 L. Viene conteggiata nel piatto anche la posta fissa che è di 10.000 L. Le regole seguite sono quelle classiche del gioco con due puntate e rilanci a oltranza. Il giocatore deve quindi solo fare le sue puntate e segnalare le carte che vuole cambiare (premendo il "RETURN" ad ogni inserimento). Al resto penserà il computer segnalando chi ha vinto e con quale combinazione. La partita viene vinta da chi riesce a mandare l'avversario al di sotto delle 10.000 L.
Per far girare questo programma il VIC deve avere un'espansione di almeno 16 Kbyte RAM e prima del caricamento è necessario dare le istruzioni POKE 43,0: POKE 44,33: POKE 8447,0 (Return). Infatti per la creazione delle figure occorre spostare l'indirizzo di inizio del programma per lasciare spazio in memoria ai nuovi e ai vecchi caratteri di stampa. Le carte del mazzo e dei giocatori sono

Foto 3 - Secondo rilancio.

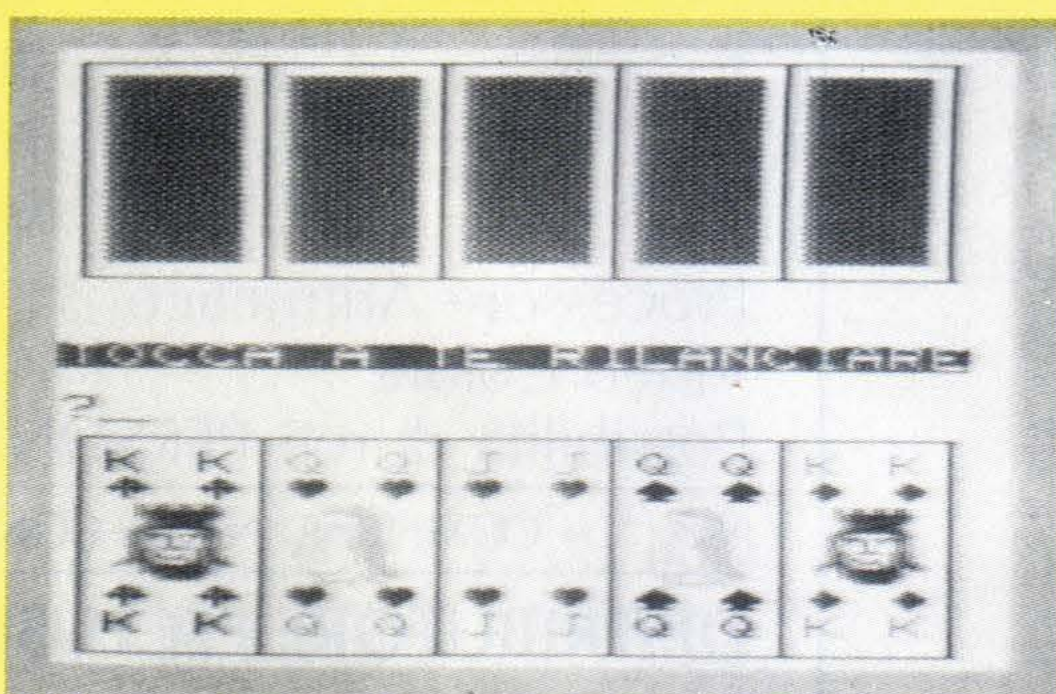



Foto 1 - Primo rilancio.

Foto 4 - Responso.

Foto 2 - Cambio carte.

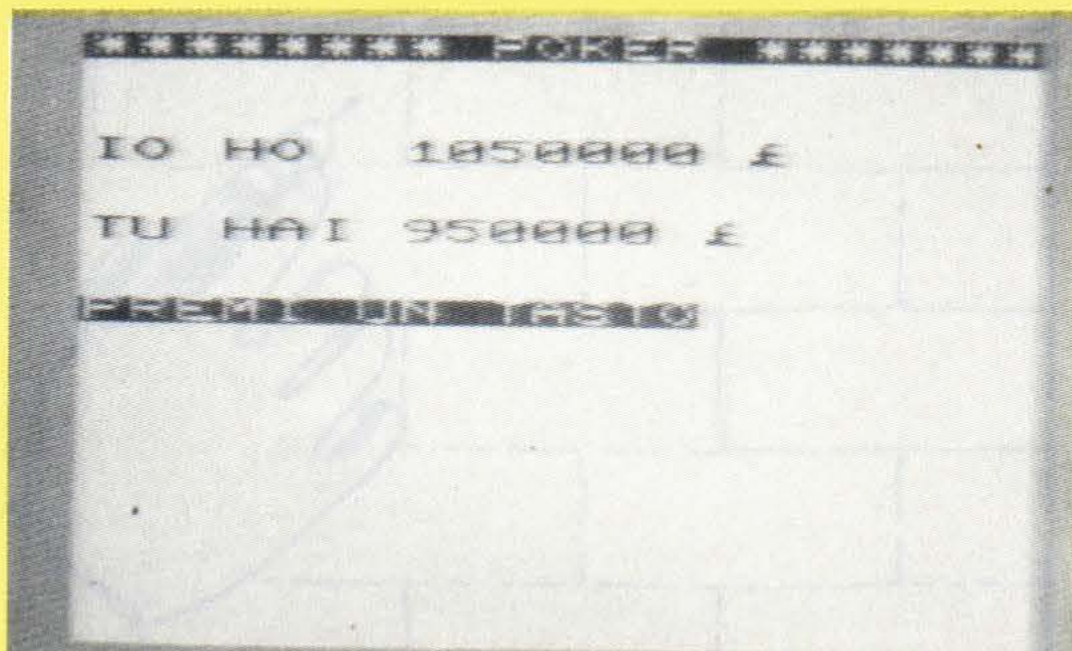



Foto 5 - Presentazione totali.

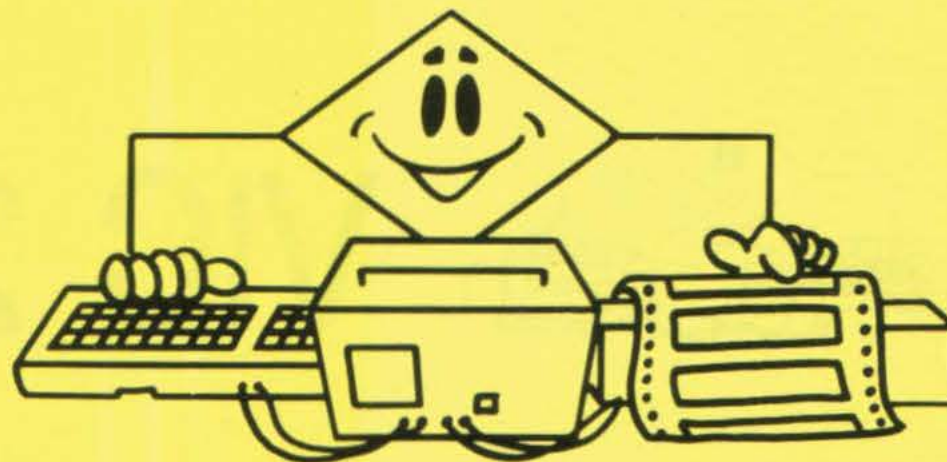
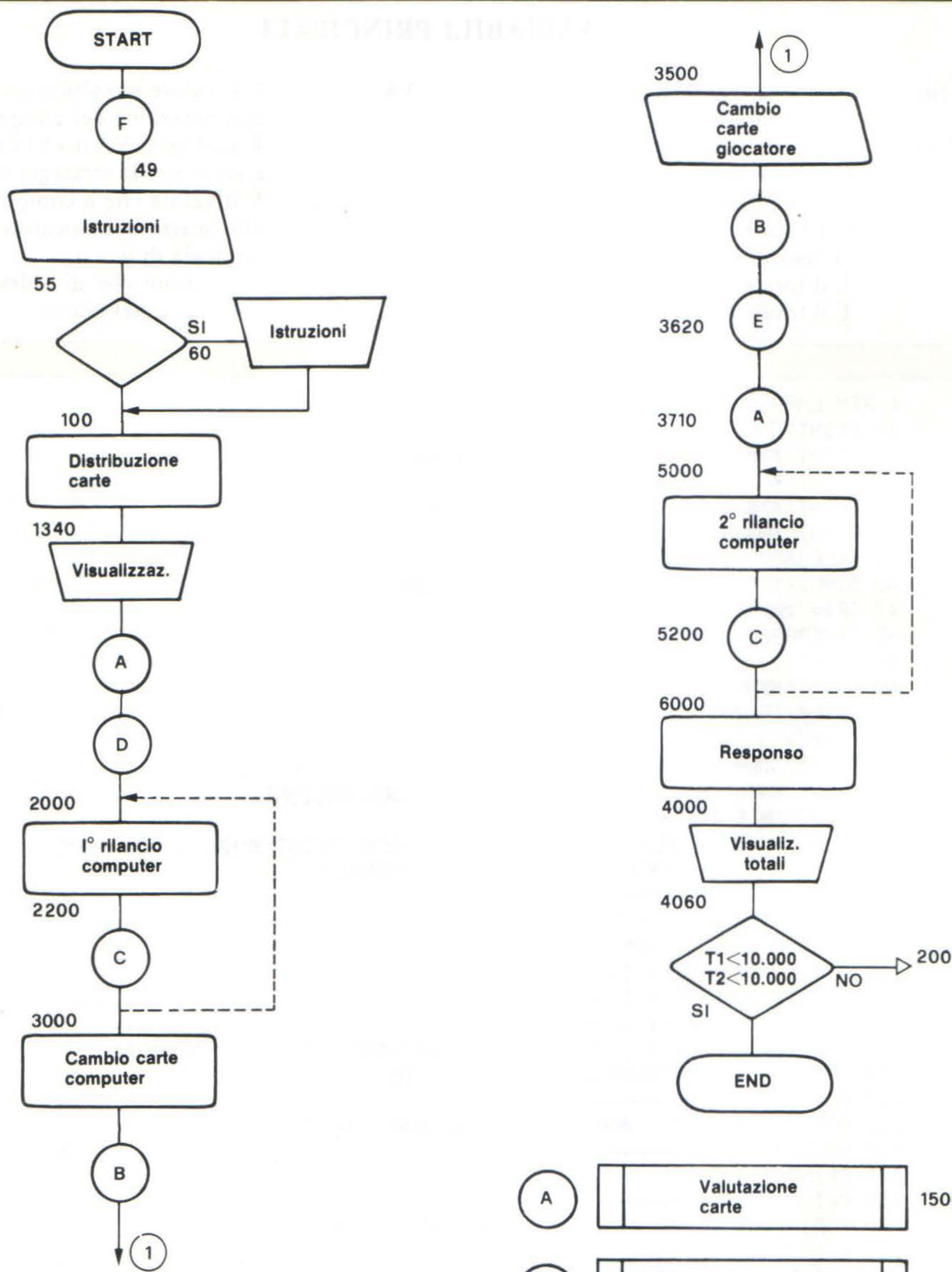
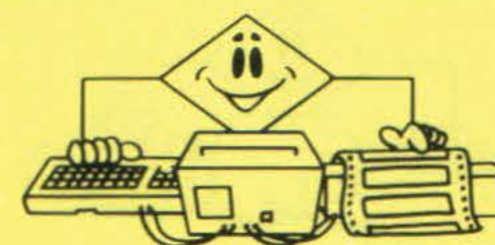



Figura 1 - La struttura, semplificata, che corrisponde al programma per giocare a Poker.

memorizzate in variabili con indice (V. linea 40). Un'idea chiara della struttura del programma può essere data dal diagramma a blocchi in figura.
Come si può vedere esso è composto da un certo numero di subroutine per le funzioni che si ripetono più spesso. La valutazione delle carte viene eseguita dando a ciascuna combinazione un valore che va da 0 per la minima, 10 per la coppia, sino ad 80 per la scala reale. A questo numero per le combinazioni: coppia, doppia, tris e full ne viene aggiunto uno che va da 1 a 6. Questo per distinguere combinazioni uguali con carte differenti. Per esempio un tris di 9 vale 31 mentre uno di RE vale 35. Infatti il tris nell'ordine suddetto vale 30 mentre il 9 che è la carta più piccola vale 1 (il re vale 5). Per la strategia del gioco del computer viene usata una scala differente con valori tanto più elevati quanto è elevata la combinazione che ha il computer stesso. Nella valutazione per il secondo rilancio viene inserito un altro numero che tiene conto dello scarto del giocatore (uno scarto di 2 ad esempio indica in genere che il giocatore ha un tris). Vi è naturalmente anche una routine che decide quando è il caso per il computer di provare un bluff.
Buona fortuna!

VARIABILI PRINCIPALI

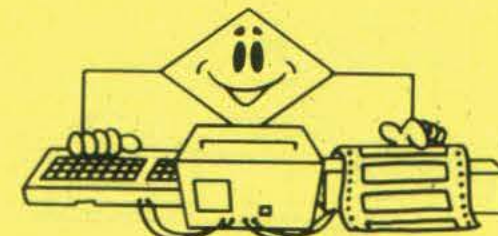
C (4,10)	Contiene le carte pescate (valore 1) e quelle da pescare (valore 0).	**VA**	È il valore numerico assegnato alla combinazione del computer.
C1 (1,5)	Contiene le carte del giocatore.	**VV**	È analogo a VA ma ha valori diversi e serve per la strategia di gioco.
C2 (1,5)	Contiene le carte del computer.		
A	È il seme della carta pescata.	**VT**	È il valore che il computer assegna allo scarto del giocatore per la sua strategia di gioco.
B	È il valore della carta pescata (dal 9 all'asso).		
T1	È il totale posseduto dal giocatore.	**PO**	È il valore che di volta in volta è contenuto nel piatto.
T2	È il totale posseduto dal computer.		

Figura 2 - Il listato BASIC.

```
0 REM EMANUELE MORELLIPERUGIA GIUGNO 1983
10 PRINT"□";
20 PRINT"■■■*********■ POKER ■■*********"
30 PRINT"■          ■♥ ♦■ ♣ ♠"
31 PRINT"■■■DI EMANUELE MORELLI   ■PERUGIA 1983"
32 PRINT"■■■■   ATTENDERE  PREGO   "
35 POKE36878,15:GOSUB15000
40 DIMC1(1,5):DIMC2(1,5):DIMC3(1,5):DIMV2(6):DIMC(4,6)
42 SP$="■■■■■■■■■■■"
43 T1=1000000:T2=1000000
44 PP$="
49 PRINT"■■■VUOI LE ISTRUZIONI ?"
50 GETA$:IFA$=""THEN50
55 IFA$<>"S"THEN100
60 PRINT"□■■♥ ♦ ♣ ♠ POKER ♠ ♣ ♦ ♥ "
65 PRINT"■■IL CAPITALE INIZIALE  E' DI ■UN MILIONE."
70 PRINT"■■LE ISTRUZIONI SONO :"
75 PRINT"■'VE' = VEDO":PRINT"■'PS' = PASSO":PRINT"■'NA' = NON APRO"
80 PRINT"■'PR' = PAROLA":PRINT"■'SE' = SERVITO"
85 PRINT"■■PUNTATA MAX £. 99000"
90 PRINT"■■■■PREMI UN TASTO"
95 GETA$:IFA$=""THEN95
100 FORI=1TO4:READS$(I):NEXT
110 FORI=1TO6:READC$(I):NEXT
111 FORI=1TO6:READF$(I):NEXT
112 FORI=0TO8:READR$(I):NEXT:FORI=1TO6:READR1$(I):NEXT:RESTORE
200 PRINT"□";:POKE36879,221:POKE36878,15
210 PRINT"■┌────┬────┬────┬────┬────┐"
220 FORI=1TO7:PRINT"|▒▒▒▒|▒▒▒▒|▒▒▒▒|▒▒▒▒|▒▒▒▒|":NEXT
230 PRINT"└────┴────┴────┴────┴────┘"
240 PRINT"■■■■■■"
250 PRINT"┌────┬────┬────┬────┬────┐"
260 FORI=1TO7:PRINT"|    |    |    |    |    |":NEXT
270 PRINT"└────┴────┴────┴────┴────┘";
320 PRINT"■■■■■■■■■■■■■■■■■■■■";
325 TE=INT(RND(0)*50):FORI=1TOTE:A=RND(0):NEXT
330 FORI=1TO5:FORT=1TO300:NEXT
340 A=INT(RND(1)*4+1):S$=S$(A)
345 B=INT(RND(1)*6+1):C$=C$(B):F$=F$(B)
365 IFC(A,B)=1THENI=I-1:GOTO1360
380 C(A,B)=1:C1(0,I)=A:C1(1,I)=B
1340 IFA<3THENPRINT"■";:GOTO1350
1345 PRINT"■";
1350 PRINTC$" "C$"■■■■"S$" "S$"■■■■"F$"■■■■";
1352 IFA<3THENPRINT"■";:GOTO1356
1353 PRINT"■";
1356 PRINTS$" "S$"■■■■"C$" "C$"TTTTTT■■";
1360 NEXT
1400 FORI=1TO5
1410 A=INT(RND(1)*4+1)
1420 B=INT(RND(1)*6+1)
1430 IFC(A,B)=1THENI=I-1:GOTO1450
1440 C(A,B)=1:C2(0,I)=A:C2(1,I)=B
1450 NEXT
1490 GOSUB1500:T=190:GOSUB9000:GOTO2000
1500 REM VALUTAZIONE
1510 FORI=1TO5
1520 V2(C2(1,I))=V2(C2(1,I))+1
```

Seguito figura 2.

```
1530 NEXT
1540 REM *COLORE-SCALA REALE
1550 CI=C2(0,1):FORI=2TO5
1555 IFNOTC2(0,I)=CITHENNC=1
1560 NEXT:IFNC=0THEN1565
1562 NC=0:GOTO1635
1565 IFV2(0)=0ORV2(6)=0THENVA=80+V2(6):VV=300+V2(6):NC=0:RETURN
1570 VA=60:VV=170:RETURN
1635 REM *POKER
1640 VA=0:FORI=1TO6
1650 IFV2(I)=4THENVA=70+I:VV=230+I:I=6
1660 NEXT:IFVA>0THENRETURN
1665 REM *TRIS-FULL-DOPPIA-COPPIA
1670 NC=0:FU=0:TR=0:FORI=1TO6
1680 IFV2(I)=3THENTR=1:V1=I
1690 IFV2(I)=2THENFU=1:NC=NC+1:V3(NC)=I
1700 NEXT
1704 IFNC=1THENV2=V3(1)
1706 IFNC=2THENV2=V3(2)
1710 IFTR=1ANDFU=1THENVA=50+V1:VV=120+V1:TR=0:NC=0:V1=0:FU=0:RETURN
1715 IFTR=1THENVA=30+V1:VV=50+V1:TR=0:NC=0:V1=0:RETURN
1720 IFFU=1THENVA=NC*10+V2:VV=NC*10+V3(1)/10+V3(2):V2=0:FU=0:NC=0:V3(1)=0:V3(2)=
0:RETURN
1730 REM *SCALA SEMPLICE
1740 IF(V2(6)=1ANDV2(1)=0)OR(V2(6)=0ANDV2(1)=1)THENVA=40+V2(6):VV=80:RETURN
1900 VA=6:VV=10:RETURN
2000 REM RILANCIO 1
2003 POKE36879,221
2005 PO=0:X1=0:X2=0:R1=0:R2=0:R3=0
2010 TU=TU+1:IFTU>1THENTU=0
2015 ON(TU+1)GOTO2050,2200
2050 IFVA<13ANDPO=0ANDTU=0THENPRINTSP$"■■NON APRO       ■":Y=7:GOTO2900
2060 IFPO>VV*5000ANDINT(RND(1)*VV)<6THENPRINTSP$"■■PASSO                ":Y=5:GOTO250
0
2072 IFPO=0ANDINT((RND(1)*VA+VA/2)*2000)<X1THEN2150
2075 IFR3=1ANDVA*5000<POTHEN2150
2076 IFR3=1ANDINT((RND(1)*VA+VA/4)*1000)<POTHEN2150
2080 X2=INT((RND(1)*VV*2+10)/10)*10000
2085 IFX2>100000THENX2=100000
2090 PRINTSP$"■"PP$;:R2=1:Y=10
2100 R$="":PRINTSP$"■■RILANCIO A ■"X2"■£    ■■":FORI=1TO3000:NEXT
2120 PRINTSP$"■                          ":GOTO2500
2150 PRINTSP$"■■VEDO                    ":FORT=1TO1000:NEXT:Y=8:R2=1:GOTO2500
2200 PRINTSP$"■■■TOCCA A TE RILANCIARE■■"
2205 K$="":GOSUB20000:K$=B$:B$=""
2210 IFLEFT$(K$,2)="VE"THENY=4:R1=1:GOTO2500
2215 IFLEFT$(K$,2)="PS"THENY=1:GOTO2500
2225 IFLEFT$(K$,2)="NA"THENY=3:GOTO2500
2226 IFVAL(K$)=0THENY=4:R1=1:GOTO2500
2230 X1=VAL(K$):R1=1:Y=9:R3=1:GOTO2500
2500 IF(Y>1ANDY<5)ORY>5ANDR1=1ANDR2=1THENPO=PO+X1+X2:X1=0:X2=0:R1=0:R2=0
2600 PK=PO:ONYGOTO2620,2000,2660,3000,2700,2000,2740,3000,2050,2200
2620 T1=T1-10000-PO:T2=T2+10000+PO:GOTO4000
2660 IFCT=1THEN4000
2665 CT=1:FORT=1TO2000:NEXT:GOTO2010
2700 T1=T1+10000+PO:T2=T2-10000-PO:GOTO4000
2740 IFCT=1THEN4000
2745 CT=1:FORT=1TO2000:NEXT:GOTO2010
2900 FORI=1TO2000:NEXT:PRINTSP$"                   ■■":GOTO2500
3000 REM CAMBIO CARTE
3005 POKE36879,219
3010 PRINTSP$PP$PP$
3020 CA=INT(VA/10)
3030 IFCA>3THENM$="■SERVITO":GOTO3500
3040 IFINT(RND(1)*16)=0ANDVA>12THENCA=4:BL=1:GOTO3030
3050 ONCAGOTO3200,3300,3400
3100 FORI=1TO5
3110 IFC2(1,I)=6THENC2(1,1)=6:C2(0,1)=C2(0,I)
3120 NEXT
3130 FORI=2TO5:GOSUB10000:C2(0,I)=A:C2(1,I)=B:NEXT
3140 M$="CAMBIO 4 CARTE":GOTO3500
3200 CB=VA-10*CA
3205 FORI=1TO5
```

Seguito figura 2.

```
3210 IFC2(1,I)=CBTHEN3230
3215 GOSUB10000
3220 C2(0,I)=A:C2(1,I)=B
3230 NEXT
3240 M$="CAMBIO 3 CARTE":GOTO3500
3300 CB=VA-10*CA
3305 IFINT(RND(1)*2)=0THEN3350
3310 FORI=1TO6:IFV2(I)=1THENSI=I:NEXT
3315 FORI=1TO5
3320 IFC2(1,I)=SITHENGOSUB10000:C2(0,I)=A:C2(1,I)=B
3325 NEXT
3330 M$="CAMBIO UNA CARTA":GOTO3500
3350 FORI=1TO5
3355 IFC2(1,I)=CBTHEN3380
3360 GOSUB10000
3365 C2(0,I)=A:C2(1,I)=B
3380 NEXT
3390 M$="CAMBIO 3 CARTE":GOTO3500
3400 CB=VA-10*CA
3405 FORI=1TO5
3410 IFC2(1,I)=CBTHEN3430
3415 GOSUB10000
3420 C2(0,I)=A:C2(1,I)=B
3430 NEXT
3440 M$="CAMBIO 2 CARTE"
3500 IFTU=0THENPRINTSP$"■■"M$:GOTO3520
3515 PRINTSP$"                "
3520 PRINTSP$"■■■■■  ■1■    ■2■    ■3■    ■4■     ■5"
3530 QS=0:Z$="":PRINTSP$"■■QUALI CAMBI ■";:LL=1:LH=4:GOSUB20000:Z$=B$:B$="":LL=0
:LH=0
3535 IFTU=1THENPRINTSP$"■■"M$:FORT=1TO1000:NEXT
3552 IFLEFT$(Z$,1)="S"ORLEFT$(Z$,1)=""THENQS=1:GOTO3620
3560 LZ=LEN(Z$)
3570 FORI=1TOLZ
3580 J=VAL(MID$(Z$,I,1))
3590 GOSUB10000
3600 C1(0,J)=A:C1(1,J)=B
3610 NEXT
3620 GOSUB8000
3700 REM VALUTAZIONE COM 2
3705 FORI=1TO6:V2(I)=0:NEXT:NC=0:VA=0
3710 GOSUB1500:VD=VA:VA=0:VZ=VV
3715 FORI=1TO6:V2(I)=0:NEXT:NC=0
3720 FORI=1TO5:C3(0,I)=C2(0,I):C3(1,I)=C2(1,I):C2(0,I)=C1(0,I):C2(1,I)=C1(1,I):N
EXT
3730 GOSUB1500:VC=VA:VX=VV
3740 R1=INT(VC/10):R2=VC-R1*10:VA=VD
3750 R1$=R$(R1)+" "
3760 IFR1=4ORR1=6ORR1=8THEN3770
3765 R1$=R1$+R1$(R2)
3770 R3=INT(VD/10):R4=VD-R3*10
3780 R2$=R$(R3)+" "
3790 IFR3=4ORR3=6ORR3=8THEN3820
3800 R2$=R2$+R1$(R4)
3810 IFQS=1THENQS=0:GOTO5000
3820 PRINT"■■■■■■■■■■■■■■■■■■■"
3821 FORI=1TO5:PRINT"   ■■■■   ■■■■   ■■■■   ■■■■   ■■■■   ■■■■   TTTTT■";:NEXT
3822 PRINT"■■■■■■■■■■■■■■■■■■■";
3825 FORI=1TO5:FORT=1TO500:NEXT
3830 IFC1(0,I)<3THENPRINT"■";:GOTO3850
3840 PRINT"■";
3850 S$=S$(C1(0,I))
3860 C$=C$(C1(1,I)):F$=F$(C1(1,I))
3880 PRINTC$" "C$"■■■■"S$" "S$"■■■■"F$"■■■■";
3882 IFC1(0,I)<3THENPRINT"■";:GOTO3886
3884 PRINT"■";
3886 PRINTS$" "S$"■■■■"C$" "C$"TTTTT■";
3900 NEXT
3910 GOTO5000
4000 FORT=1TO2000:NEXT
4030 PRINT"■■********* POKER *******"
4040 PRINT"■■IO HO "T2"£"
4050 PRINT"■■TU HAI"T1"£"
4052 T=128:GOSUB9000
```

```
4055 IFCT=1THENCT=0:TU=TU+1:IFTU>1THENTU=0
4060 IFT1<10000THENPRINT"HO VINTO":T=230:GOSUB9100::GOTO4500
4070 IFT2<10000THENPRINT"HAI VINTO":T=160:GOSUB9100:GOTO4500
4100 PRINT"PREMI UN TASTO":POKE198,0
4110 GETA$:IFA$=""THEN4110
4120 FORI=1TO6:V2(I)=0:NEXT:PO=0
4130 FORI=1TO4:FORJ=1TO5:C(I,J)=0:NEXT:NEXT:GOTO200
4500 PRINT"VUOI RIPROVARE (S/N) ?"
4510 GETA$:IFA$=""THEN4510
4520 IFA$="S"THENPRINT"";:RUN
4530 POKE36879,27:PRINT"";:END
5000 REM RILANCIO 2
5002 PRINTSP$PP$PP$"                    "
5003 POKE36879,223
5005 X1=0:X2=0:R1=0:R2=0:R3=0
5010 ON(TU+1)GOTO5060,5200
5060 IFPO-PK=0ANDTU=0ANDINT(RND(1)*10)=0THEN5400
5062 IFPO-PK=0ANDTU=0ANDINT(RND(1)*VZ)<8+VS/2THEN5400
5063 IFBL=1ORINT(RND(1)*10)=0THENBL=0:X2=100000:GOTO5090
5065 IFPO>VZ*8000ANDINT(RND(1)*VZ)<8+VS/2ANDY<>2THEN5450
5070 IFVT>VZAND(PO-PK)=0ANDY<>2ANDINT(RND(1)*3)=0THEN5450
5072 IFPO-PK=0ANDINT((RND(1)*VZ+VZ/2)*2000)<X1THEN5150
5075 IFR3=1ANDVZ*5000<(PO-PK)THEN5150
5076 IFR3=1ANDINT((RND(1)*VZ+VZ/2-VS)*4000)<(PO-PK)THEN5150
5080 X2=INT((RND(1)*VZ+VZ/4)/(10-INT(RND(1)*4)))*10000
5085 IFX2>100000THENX2=100000
5086 IFX2<10000THENX2=10000
5090 PRINTSP$""PP$;:R2=1:Y=10:NV=1
5100 R$="":PRINTSP$"RILANCIO A "X2"£    ":FORI=1TO3000:NEXT
```

Seguito figura 2.

VIC 20

Seguito figura 2.

```
5120 PRINTSP$"█                    ":GOTO5500
5150 PRINTSP$"█VEDO                ":FORT=1TO1000:NEXT:Y=8:R2=1:GOTO5500
5200 PRINTSP$"███TOCCA A TE RILANCIARE██"
5205 K$="":GOSUB20000:K$=B$:B$=""
5210 IFLEFT$(K$,2)="VE"THENY=4:R1=1:GOTO5500
5215 IFLEFT$(K$,2)="PS"THENY=1:GOTO5500
5220 IFLEFT$(K$,2)="PR"THENY=2:GOTO5500
5225 IFVAL(K$)=0THENY=4:R1=1:GOTO5500
5230 X1=VAL(K$):R1=1:Y=9:R3=1:GOTO5500
5400 PRINTSP$"█PAROLA          ":Y=6:FORT=1TO2000:NEXT:PRINTSP$PP$:GOTO5500
5450 PRINTSP$"█PASSO           ":Y=5
5500 IF(Y>1ANDY<5)ORY>5ANDR1=1ANDR2=1THENPO=PO+X1+X2:X1=0:X2=0:R1=0:R2=0
5600 ONYGOTO5620,5640,10,6000,5700,5720,10,6000,5060,5200
5620 T1=T1-10000-PO:T2=T2+10000+PO:GOTO4000
5640 IFCT=1THENCT=0:GOTO4000
5645 CT=1:TU=0:FORT=1TO2000:NEXT:GOTO5010
5700 T1=T1+10000+PO:T2=T2-10000-PO:GOTO4000
5720 IFCT=1THENCT=0:GOTO4000
5725 CT=1:TU=1:FORT=1TO2000:NEXT:GOTO5010
6000 REM RESPONSO
6005 POKE36879,222
6010 PRINT"███";:FORI=1TO5:PRINT"  ████  ████  ████  ████  ████  ████  T
TTTT█";:NEXT
6020 PRINT"███";:FORI=1TO5:FORT=1TO500:NEXT
6030 IFC3(0,I)<3THENPRINT"█";:GOTO6050
6040 PRINT"█";
6050 S$=S$(C3(0,I))
6070 C$=C$(C3(1,I)):F$=F$(C3(1,I))
6080 PRINTC$" "C$"████"S$" "S$"████"F$"████";
6082 IFC3(0,I)<3THENPRINT"█";:GOTO6086
6084 PRINT"█";
6086 PRINTS$" "S$"████"C$" "C$"TTTTTT█";
6120 NEXT
6130 PRINTSP$PP$PP$PP$SP$"IO HO "R2$" TU HAI "R1$
6140 IFVX=VZTHENPRINT"███PARITA'":RE=1:GOTO7000
6150 IFVZ<VXTHENPRINT"███HAI VINTO":T1=T1+PO+10000:T2=T2-PO-10000:RE=2:GOTO7000
6160 IFVZ>VXTHENPRINT"███HO VINTO":T1=T1-PO-10000:T2=T2+PO+10000:RE=3:GOTO7000
7000 PRINTSP$"██████───────────────"
7010 ONREGOSUB9200,9200,9300
7030 PRINTSP$"██████ PREMI UN TASTO"
7040 GETA$:IFA$=""THEN7040
7050 GOTO4030
8000 IFQS=1THENVS=12-INT(RND(1)*4):VT=VS*4:RETURN
8010 ONLZGOTO8020,8030,8040,8050
8020 VS=5:VT=42:RETURN
8030 VS=8:VT=52:RETURN
8040 VS=3:VT=22:RETURN
8050 VS=1:VT=12:RETURN
9000 POKE36876,T:POKE36879,29:FORT=1TO250:NEXT:POKE36876,0:POKE36879,221:RETURN
9100 FORI=1TO5:POKE36876,T:POKE36879,218:FORIT=1TO200:NEXT:POKE36876,0:POKE36879
,217
9150 FORIT=1TO200:NEXT:NEXT:POKE36879,221:RETURN
9200 FORI=1TO2:POKE36876,240:POKE36879,29:FORIT=1TO200:NEXT:POKE36876,0:POKE3687
9,26
9220 FORIT=1TO200:NEXT:NEXT:POKE36879,221:RETURN
9300 FORI=1TO2:POKE36876,150:POKE36879,28:FORIT=1TO200:NEXT:POKE36876,0:POKE3687
9,24
9320 FORIT=1TO200:NEXT:NEXT:POKE36879,221:RETURN
10000 A=INT(RND(1)*4+1)
10010 B=INT(RND(1)*6+1)
10020 IFC(A,B)=1THEN10000
10030 C(A,B)=1:RETURN
15000 FORI=0TO1751:POKE6144+I,PEEK(32768+I):NEXT
15010 FORI=0TO215:READA:POKE7896+I,A:NEXT
15020 POKE36869,206:RETURN
20000 PRINT"?_         ████████";:I=0
20010 GETA$:IFA$=""THEN20010
20020 IFA$=CHR$(13)THEN20060
20030 IFA$=CHR$(20)ANDI>0THENI=I-1:PRINT"█ ██_";:B$=LEFT$(B$,I):GOTO20010
20040 AS=ASC(A$):IFNOT((AS>47+LLANDAS<58-LH)OR(AS>64ANDAS<91))THEN20010
20045 IFLH=1ANDI>3THEN20010
20047 IFI>4THEN20010
20050 PRINT"█"A$"_";:I=I+1:B$=B$+A$:GOTO20010
```

Seguito figura 2.

```
20060 PRINT"▌ ";:FORI=1TOLEN(B$)+1:PRINT"▌";:NEXT:PRINT"▓"B$"▀";:RETURN
30000 DATA27,13,6,3,1,3,7,7,126,189,0,255,255,255,248,241,216,176,96,192,128,128
,64,64
30010 DATA7,7,7,15,15,31,63,127,227,228,224,224,192,192,129,0,64,32,16,16,96,64,
192,64
30020 DATA254,6,2,12,31,63,127,127,0,32,31,4,252,252,254,255,32,32,192,0,0,0,0,0
30030 DATA0,1,3,4,8,9,10,10,254,255,255,15,7,71,55,39,0,0,128,128,192,192,192,19
2
30040 DATA8,8,17,18,18,19,16,20,135,143,15,15,15,15,7,135,192,192,224,224,224,22
4,224,224
30050 DATA19,16,16,8,7,2,3,6,7,13,17,33,194,7,253,170,224,240,240,248,124,188,19
2,96
30060 DATA3,3,3,3,1,0,1,3,51,51,255,255,255,255,255,255,48,48,240,240,224,192,19
30070 DATA7,6,6,6,6,7,7,7,0,227,73,8,8,28,0,34,112,176,48,48,48,112,112,112
30080 DATA15,29,49,1,1,0,0,0,28,129,255,255,255,127,63,0,120,220,198,192,192,128
30,0
30090 DATA"♥","♦","♣","♠"
30005 DATA9,D,J,Q,K,A
30010 DATA"▓▓▓▓▌","▓▓▓▓!"
30015 DATA"▓▓▓ ▓▓▓▓▓▓ ▓▓▓▓▄▀▀"
40020 DATA"▓▓▓ ▒▓▓▓▓ ▓▓▓▓▓!|├▓"
40025 DATA"▓▓┤▄▓▓▓┌┬▓▓▓H||▓"
40030 DATA"    ▓▓▓▓ A ▓▓▓▓    "
40060 DATAMINIMO,COPPIA,DOPPIA,TRIS,SCALA SEMPLICE
40070 DATAFULL,COLORE,POKER,SCALA REALE
40080 DATADI NOVE,DI DIECI,DI JACK,DI DONNE,DI RE,DI ASSI
```

SIMBOLI GRAFICI

```
▓ = SHIFT + CLR = CHR$(147)
▓ = CLR = CHR$(19)
▓ = CRSR ↑ = CHR$(17)
▓ = SHIFT + CRSR  = CHR$(145)
▌ = CRSR ← = CHR$(29)
▌ = SHIFT + CRSR ← = CHR$(145)
▓ = SHIFT + RVS ON = CHR$(18)
▀ = SHIFT + RVS OFF = CHR$(146)
■ = CTRL + BLK = CHR$(144)
▓ = CTRL + RED = CHR$(28)
▓ = CTRL + PUR = CHR$(156)
▓ = CTRL + GRN = CHR$(30)
▓ = CTRL + BLU = CHR$(31)
▓ = CTRL + YEL = CHR$(158)
```

A proposito del Personal Data Base per Olivetti M20

Le numerosissime richieste del package PDB per M20 ci hanno creato qualche piccolo problema anche in relazione alla scelta di rendere più flessibile l'interfaccia utente. Preghiamo pertanto tutti coloro che ne hanno fatto richiesta di pazientare ancora qualche tempo.

AGE 84: memorizzare gli amici

Sembra quasi che stia dilagando la mania di sviluppare programmi di archiviazione dati con il famigerato ZX80/81, forse perchè è una delle sue maggiori lacune o forse per farla in barba a Clive che non ha messo le istruzioni per la gestione dei file nel suo BASIC.
Comunque, scherzi a parte, vi propongo un programma che gestisce un micro archivio; si tratta di una agenda telefonica completa di indirizzi con 135 nominativi memorizzabili.

a cura della **Redazione**

La ricerca dei record avviene su più chiavi quali cognome, nome (o entrambi), indirizzo e numero di telefono.
Dimenticavo, il programma gira su uno ZX80 8 Kbyte ROM e 16 Kbyte RAM, non è previsto quindi l'uso della funzione SLOW.

Il programma

Il corpo del programma è costituito da sei sottoprogrammi chiamati su opzione dal programma principale.
A discapito della strutturalità del programma, dandogli una sbirciatina, si notano diversi GOTO che fanno sembrare lo stesso alquanto "spaghettaro".
Purtroppo con il BASIC, specie quello Sinclair, non si può fare molto di meglio.
Osservando bene i GOTO, essi servono per semistrutturare il programma simulando l'istruzione ELSE non presente nel BASIC Sinclair, a creare dei cicli per l'aggiunta di spazi bianchi ai campi di lunghezza inferiore a quella stabilita (SUB di ricerca) e per ottimizzare l'occupazione di memoria allacciando più parti di programma.
Il programma principale inizia con il dimensionamento dei sei settori stringa che conterranno i campi dei record da memorizzare, segue l'inizializzazione della variabile A che conta i record presenti in memoria e punta sempre all'ultimo record memorizzato.
Le stringhe L$, D$, O$, P$ e Q$, sono usate per la mascheratura dello schermo durante l'input e l'output dei dati.
La linea 130 contiene i caratteri grafici che servono a disegnare un telefono sullo schermo per la presentazione del programma.
Abbiamo infine la stampa delle opzioni ed il GOSUB Z ★ 1000 che fa accedere alla SUB prescelta.
Per avere chiaro il funzionamento del programma, occorre studiare prima le due SUB principali cioè la SUB di inserimento e la SUB di ricerca.
La SUB di inserimento inizia con l'incremento unitario della variabile A.
Se essa è superiore a 135 si ha il ritorno al menu per delle altre operazioni che non siano di inserimento, altrimenti si prosegue con l'input dei dati.
La linea 2220 ci domanda se si desidera correggere l'inserimento, se la risposta è "S" si ripete la fase di input dati, altrimenti, rispondendo con qualsiasi carattere od il solo NEW LINE, si prosegue con la linea 2270 che ci chiede invece se si desidera annullare l'inserimento.
Rispondendo con "N" o con NEW LINE, il nominativo viene memorizzato e si ritorna al menu; rispondendo con "S", (o qualsiasi carattere eccetto la "N"), si decrementa la variabile A di uno, quindi il nominativo appena inserito non è più puntato e con un nuovo inserimento ci si scrive sopra.
La SUB di ricerca è più lunga e più complessa delle altre perchè permette di ricercare un record con 1 o 2 dei suoi campi, (indicati come opzioni, linea 3050).
Per quanto riguarda la variabile C inizializzata nella linea 3000 si vedrà in seguito a cosa serve.
L'algoritmo di ricerca è quasi identico per tutte le opzioni tranne qualche lieve differenza.
In pratica si esegue una scansione sequenziale del vettore contenente il campo scelto (con la prima opzione la ricerca è parallela nei campi cognome e nome).
Descriviamo ora come avviene la ricerca con la prima opzione:
il programma ci chiede di inserire il cognome, se è più lungo di 15 caratteri non lo accetta e lo richiede (linea 3110); la linea 3115 controlla se questo è uguale a quindici caratteri, se è affermativo si prosegue con la linea 3130, altrimenti si aggiungono tanti spazi bianchi quanto ne servono per arrivare alla lunghezza giusta e così si salta alla linea 3130 che con la linea 3135 vuole inserito il nome.
Si esegue lo stesso controllo, se è il caso si aggiungono gli spazi bianchi e quando il nome è lungo 10 caratteri si inizia la ricerca parallela con un ciclo da 1 ad A (ultimo record).
Se si trova il nominativo, si salta alla linea 3600 che visualizza sullo schermo l'output del record

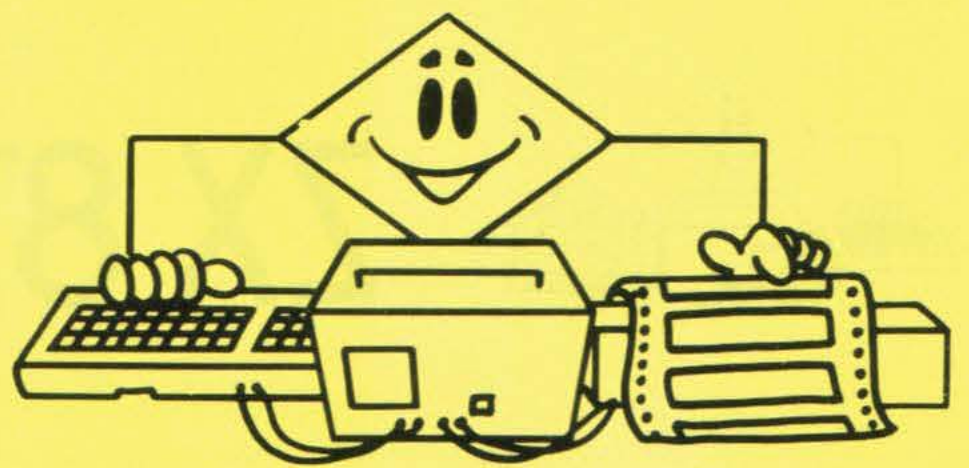



Figura 1 - I diagrammi di flusso che rappresentano le routine di cui è costituito il programma.

INIZIO
Dimens. vettori
iniz. variabili
Stampa opzioni
INPUT Z
Z=1 si SUB lista
Z=2 si SUB inserimento
Z=3 si SUB ricerca
Z=4 si SUB cancellazione
Z=5 si SUB azzeramento
Z=6 si SUB salvataggio

SUB lista
lista ribrica
premi N/L, M=menu
INPUT Y$
Y$ = "M" si RETURN
K=1
K>A si RETURN
SUB visualizzazione
Y$ = "M" si RETURN
K=K + 1

SUB inserimento
A=A + 1
A>135 si memoria piena
inserimento dati
premi N/L, M=menu
INPUT Y$
Y$ = "M" si RETURN
stampa campi
INPUT C$(A)
PRINT C$(A)
INPUT N$(A)
PRINT N$(A)
INPUT I$(A)
PRINT I$(A)
INPUT A$(A)
PRINT A$(A)
INPUT T$(A)
PRINT T$(A)
INPUT F$(A)
PRINT F$(A)
vuoi correggere? S/N
INPUT Y$
Y$ = "S" si
vuoi annullare? S/N
INPUT Y$
Y$ = "S" si A=A - 1 RETURN
RETURN

cercato altrimenti, a fine ciclo (dopo NEXT K), si salta alla linea 3660 che ci informa dell'inesistenza del record in memoria.

Dato che in una agenda si possono trovare delle persone con lo stesso cognome o lo stesso nome, la routine di ricerca con le opzioni 2 e 3 è leggermente diversa.
Invece di andare alla visualizzazione record con un GOTO, va con un GOSUB (linee 3270 - 3370) quindi, dopo aver incrementato la variabile C di 1 (linea 3630), per indicare l'avvenuto passaggio, si ritorna al ciclo (3640 RETURN).
Se esiste un altro campo identico, si ripete il passaggio.
Alla fine del ciclo si salta alla linea 3650 che controlla il valore di C, se esso è maggiore di 0 significa che si è fatto almeno un passaggio, quindi si ritorna al menu.

Seguito figura 1.

SUB azzeramento
vuoi veramente azzerare? S/N
INPUT Y$
Y$ = "M" — si → RETURN
RUN 10

SUB salvataggio
salvataggio dati premu N/L, M= menu
INPUT Y$
Y$ = "M" — si → RETURN
premi N/L quando se pronto
INPUT Y$
SAVE "rubrica" GOTO 130

(1) → LEN (B$) = 15 / LEN (E$) = 10 — si → K=1; → B$ =B$+"b" / E$ =E$+"b"; K>A — si → (8); B$ =C$ (K) / E$ =E$ (K) — si → (6); K=K +1

(2) → LEN (B$) = 15 — si → K=1; → B$ =B$+"b"; K>A — si → (7); B$ =C$ (K) — si → GSB 6; K=K +1

(3) → LEN (B$) = 10 — si → K=1; → B$ =B$+"b"; K>A — si → (7); B$ =N$ (K) — si → GSB 6; K=K +1

(4) → LEN (B$) = 20 — si → K=1; → B$ =B$+"b"; K>A — si → (8); B$ =I$ (K) — si → (6); K=K +1

(5) → LEN (B$) = 12 — si → K=1; → B$ =B$+"b"; K>A — si → (8); B$ =F$ (K) — si → (6); K=K +1

(6)
SUB visualizzazione
stampa record
INPUT Y$
C= + 1
RETURN

(7)
C>0 — si → RETURN
(8) →
nominativo non presente in memoria
RETURN

SUB cancellazione
cancellazione record premi N/L, M=menu
INPUT Y$
Y$ = "M" — si → RETURN
quale nominativo cancellare?
SUB ricerca *
K>A — si → RETURN
K=A — si →
C$(K) = C$(A)
N$(K) = N$(A)
I$(K) = I$(A)
A$(K) = A$(A)
T$(K) = T$(A)
F$(K) = F$(A)
C$(A) = ""
N$(A) = ""
I$(A) = ""
A$(A) = ""
T$(A) = ""
F$(A) = ""
nominativo cancellato
RETURN

SUB ricerca
C=0
ricerca record premi N/L, M=menu
INPUT Y$
Y$ = "M" — si → RETURN
stampa messaggio per ricerca
INPUT Z
Z=1 — si * → INPUT B$; E$ → LEN (B$) > 15 / LEN (E$) > 10 — si; → (1)
Z=2 — si → INPUT B$ → LEN (B$) > 15 — si; → (2)
Z=3 — si → INPUT B$ → LEN (B$) > 10 — si; → (3)
Z=4 — si → INPUT B$ → LEN (B$) > 20 — si; → (4)
Z=5 — si → INPUT B$ → LEN (B$) > 12 — si; → (5)

ZX 81

Se il record non è presente, non avviene nessun passaggio per la routine di visualizzazione, quindi C rimane O e dopo il test della linea 3650 si ha la stampa del messaggio: "nominativo non presente in memoria".
La ricerca con i campi telefono ed indirizzo è la stessa della prima opzione, ma avviene separatamente nell'uno o nell'altro vettore.
La SUB lista esegue un ciclo, (linea 1030), e visualizza, (GOSUB 3600), tutti i record presenti in ordine di inserimento.
La SUB di cancellazione ci domanda quale nominativo cancellare, dopodichè prosegue (linea 4050) alla SUB di ricerca 1 opzione.
Se il record è presente si ha la sua visualizzazione ed il ritorno alla SUB di cancellazione (previa pressione di N/L), che procede cancellando il record mediante la sovrascrittura dell'ultimo record, (linee 4080 - 4130) e la cancellazione dell'ultimo record già memorizzato al posto di quello eliminato, (linee 4140 - 4190).
La variabile K all'uscita del ciclo FOR-NEXT di ricerca è uguale ad A+1, quindi se il record non è presente, dopo il messaggio ".. bla bla ..", ritornando alla SUB di cancellazione con il test della linea 4060, si ha il ritorno al menu.
La SUB di azzeramento, dopo la risposta affermativa, esegue un RUN 10 che, come tutti sanno, azzera le variabili del programma.
La SUB di salvataggio esegue da programma il SAVE "RUBRICA".

Nota

Affinchè il programma giri senza inghippi si deve inizializzare la variabile C.
Occorre, quindi, solo per la prima volta, inserire almeno un nominativo e poi cercarlo; fatto ciò si può tranquillamento continuare ad utilizzare il programma, registrandolo ogni qual volta si inseriscono nuovi nominativi.

Il listato

Gli spazi bianchi sono stati rappresentati con il simbolo "b", ma ove era necessario evidenziarli, (in tal caso bisogna tener conto solo di loro, non quelli della macchina da scrivere), altrove sarà il vostro buon senso a guidarvi.
I caratteri in campo inverso sono stati rappresentati chiusi dentro un rettangolino.
I simboli della linea 130 corrispondono ai seguenti caratteri grafici:
"X" = CHR$ 8;
"ˆ" = CHR$ 10;
"★" = CHR$ 151;
"—" = CHR$ 9.
Gli accenti e gli apostrofi si realizzano con i caratteri grafici 1, 2 o 130.

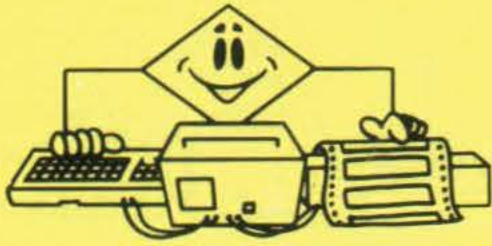
Le modifiche

Il programma si può modificare per contenere altro genere di dati, basta cambiare l'intestazione dei campi e la loro lunghezza (ovviamente in relazione alla memoria disponibile).
La SUB lista si può modificare per ottenere la lista dell'agenda con la ZX Printer anzichè sullo schermo.
Chi avesse necessità di più di 135 nominativi può togliere alcune linee di programma non indispensabili, semplificare l'output video-grafico, accorciare qualche campo per poter aumentare così il numero di record memorizzabili.

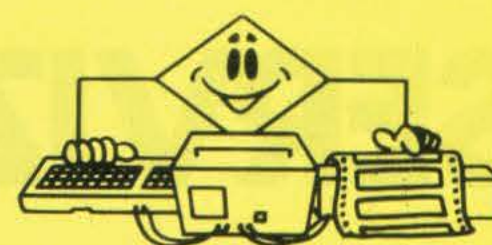
REMark

10-120	Dimensionamento vettori stringa e inizializzazione variabili.
130-200	Presentazione agenda e menu.
1000-1070	SUB lista.
2000-2050	Inizio SUB inserimento con controllo di A, se > 135 RETURN.
2100-2220	Inserimento dati.
2220-2310	Editing inserimento.
3000-3040	Inizio SUB ricerca, iniz. di C e menu di ricerca.
3100-3175	Ricerca 1ª opzione.
3200-3290	Ricerca 2ª opzione.
3300-3390	Ricerca 3ª opzione.
3400-3490	Ricerca 4ª opzione.
3500-3590	Ricerca 5ª opzione.
3600-3640	Visualizzazione record.
3650-3700	Controllo di C, se > 0 RETURN, altrimenti stampa nominativo non presente in memoria.
4000-4050	Inizio SUB cancellazione e salto alla SUB ricerca.
4060-4070	Se K>A il nominativo non è presente, se K=A bisogna cancellare solo l'ultimo record.
4080-4250	Cancellazione record.
5000-5050	SUB azzeramento.
6000-6080	SUB salvataggio.

VARIABILI

C$ (135,15)	Vettore contenente il campo cognome.
N$ (135,10)	Vettore contenente il campo nome.
I$ (135,20)	Vettore contenente il campo indirizzo.
A$ (135,5)	Vettore contenente il campo C.A.P.
T$ (135,10)	Vettore contenente il campo città.
F$ (135,12)	Vettore contenente il campo telefono.
A	Conta i record memorizzati.
C	Conta i passaggi per la routine di visualizzazione.
L$	Stringa: = "cognome".
D$	Stringa: = "nome".
O$	Stringa: = "indirizzo".
P$	Stringa: = "cap bbbbbbb città".
Q$	Stringa: = "telefono".
K	Variabili di ciclo.
Z	Variabili di input.
Y$	Variabili di input.
B$	Variabili di input.
E$	Variabili di input.

ZX 81

Listato 1 - Programma di compattazione.

```
  10 DIM C$(135,15)
  20 DIM N$(135,10)
  30 DIM I$(135,20)
  40 DIM A$(135,5)
  50 DIM T$(135,10)
  60 DIM F$(135,12)
  70 LET A=0
  80 LET L$="COGNOME"
  90 LET D$="NOME"
 100 LET O$="INDIRIZZO"
 110 LET P$="CAP      CITTA'"
 120 LET Q$="TELEFONO"
 130 PRINT AT 4,8;"[illegible]";AT 5,7;"[illegible]";AT 6,6;"[illegible]";AT 7,5;"[illegible]";AT 8,9;"[illegible]";AT 9,8;"[illegible]";AT 10,7;"[illegible]";AT 11,6;"[illegible]";AT 12,5;"[illegible]";AT 13,4;"[illegible]";AT 14,4;"[illegible]";AT 20,4;"RUBRICA  TELEFONICA"
 140 PAUSE 250
 150 POKE 16437,255
 160 CLS
 170 PRINT AT 4,2;"1 - LISTA";AT 6,2;"2 - INSERIMENTO";AT 8,2;"3 - RICERCA";AT 10,2;"4 - CANCELLAZIONE";AT 12,2;"5 - AZZERAMENTO";AT 14,2;"6 - SALVATAGGIO";AT 20,6;"SCEGLI ?"
 180 INPUT Z
 190 IF Z>6 OR Z<1 THEN GOTO 180
 200 CLS
 210 GOSUB Z*1000
 220 GOTO 160
1000 PRINT AT 4,7;"LISTA RUBRICA";AT 10,4;"PREMI<N/L>, <M>=MENU"
1010 INPUT Y$
1020 IF Y$="M" THEN RETURN
1030 FOR K=1 TO A
1040 GOSUB 3600
1050 IF Y$="M" THEN RETURN
1060 NEXT K
1070 RETURN
2000 LET A=A+1
2010 IF A<135 THEN GOTO 2060
2020 PRINT AT 5,2;"SPIACENTE, MEMORIA PIENA"
2030 PAUSE 125
2040 POKE 16437,255
2050 GOTO 2300
2060 PRINT AT 4,5;"INSERIMENTO DATI";AT 10,3;"PREMI <N/L>, <M>=MENU"
2070 INPUT Y$
2080 IF Y$="M" THEN GOTO 2300
2090 CLS
2100 PRINT AT 2,1;"INSERISCI I DATI COME SEGUE:";AT 6,1;L$;AT 9,1;D$;AT 12,1;O$;AT 15,1;P$;AT 18,1;Q$
2110 INPUT C$(A)
2120 PRINT AT 6,9;C$(A)
2130 INPUT N$(A)
2140 PRINT AT 9,6;N$(A)
2150 INPUT I$(A)
2160 PRINT AT 12,11;I$(A)
2170 INPUT A$(A)
2180 PRINT AT 15,5;A$(A)
2190 INPUT T$(A)
2200 PRINT AT 15,18;T$(A)
2210 INPUT F$(A)
2220 PRINT AT 18,10;F$(A);AT 20,1;"VUOI CORREGGERE ? <S/N>"
2230 INPUT Y$
2240 IF Y$<>"S" THEN GOTO 2270
2250 CLS
2260 GOTO 2100
2270 PRINT AT 21,1;"VUOI ANNULLARE ?  <S/N>"
2280 INPUT Y$
2290 IF Y$="N" OR Y$="" THEN RETURN
2300 LET A=A-1
2310 RETURN
3000 LET C=0
3010 PRINT AT 4,5;"RICERCA RECORD";AT 10,2;"PREMI <N/L>,  <M>=MENU"
3020 INPUT Y$
3030 IF Y$="M" THEN RETURN
3040 CLS
3050 PRINT AT 4,4;"RICERCA CON:";AT 7,4;"1 - COGNOME E NOME";AT 9,4;"2 - COGNOME";AT 11,4;"3 - NOME";AT 13,4;"4 - INDIRIZZO";AT 15,4;"5 - TELEFONO";AT 21,8;"SCEGLI ?"
3060 INPUT Z
3070 IF Z<1 OR Z>5 THEN GOTO 3060
3080 CLS
3090 GOTO Z*100+3000
3100 PRINT AT 5,2;"INSERISCI IL COGNOME"
3105 INPUT B$
3110 IF LEN (B$)>15 THEN GOTO 3105
3115 IF LEN (B$)=15 THEN GOTO 3130
3120 LET B$=B$+" "
3125 GOTO 3115
3130 PRINT AT 7,2;"INSERISCI IL NOME"
3135 INPUT E$
3140 IF LEN (E$)>10 THEN GOTO 3135
3145 IF LEN (E$)=10 THEN GOTO 3160
3150 LET E$=E$+" "
3155 GOTO 3145
3160 FOR K=1 TO A
3165 IF C$(K)=B$ AND N$(K)=E$ THEN GOTO 3600
3170 NEXT K
3175 GOTO 3660
3200 PRINT AT 5,2;"INSERISCI IL COGNOME"
3210 INPUT B$
3220 IF LEN (B$)>15 THEN GOTO 3210
3230 IF LEN (B$)=15 THEN GOTO 3260
3240 LET B$=B$+" "
3250 GOTO 3230
3260 FOR K=1 TO A
3270 IF C$(K)=B$ THEN GOSUB 3600
3280 NEXT K
3290 GOTO 3650
3300 PRINT AT 5,2;"INSERISCI IL NOME"
3310 INPUT B$
3320 IF LEN (B$)>10 THEN GOTO 3310
3330 IF LEN (B$)=10 THEN GOTO 3360
3340 LET B$=B$+" "
3350 GOTO 3330
3360 FOR K=1 TO A
3370 IF N$(K)=B$ THEN GOSUB 3600
3380 NEXT K
3390 GOTO 3650
3400 PRINT AT 5,2;"INSERISCI L*INDIRIZZO"
3410 INPUT B$
3420 IF LEN (B$)>20 THEN GOTO 3410
3430 IF LEN (B$)=20 THEN GOTO 3460
3440 LET B$=B$+" "
3450 GOTO 3430
3460 FOR K=1 TO A
3470 IF I$(K)=B$ THEN GOTO 3600
3480 NEXT K
3490 GOTO 3660
3500 PRINT AT 5,2;"INSERISCI IL NR. DI TELEFONO"
3510 INPUT B$
3520 IF LEN (B$)>12 THEN GOTO 3510
3530 IF LEN (B$)=12 THEN GOTO 3560
3540 LET B$=B$+" "
3550 GOTO 3530
3560 FOR K=1 TO A
3570 IF F$(K)=B$ THEN GOTO 3600
3580 NEXT K
3590 GOTO 3660
```

Seguito listato programma di compattazione.

```
3600 CLS
3610 PRINT AT 4,2;L$;" ";C$(K);A
T 6,2;D$;" ";N$(K);AT 8,2;O$;" "
;I$(K);AT 10,2;P$;AT 10,6;A$(K);
AT 10,19;T$(K);AT 12,2;Q$;" ";F$
(K);AT 20,1;"PER CONTINUARE PREM
I <N/L>"
3620 INPUT Y$
3630 LET C=C+1
3640 RETURN
3650 IF C>0 THEN RETURN
3660 CLS
3670 PRINT AT 5,1;"NOMINATIVO NO
N PRESENTE";AT 8,1;"IN MEMORIA"
3680 PAUSE 125
3690 POKE 16437,255
3700 RETURN
4000 PRINT AT 4,2;"CANCELLAZION
E RECORD";AT 10,2;"PREMI <N/L>,
 <M>=MENU"
4010 INPUT Y$
4020 IF Y$="M" THEN RETURN
4030 CLS
4040 PRINT AT 3,2;"QUALE NOMINAT
IVO CANCELLARE ?"
4050 GOSUB 3100
4060 IF K>A THEN RETURN
4070 IF K=A THEN GOTO 4140
4080 LET C$(K)=C$(A)
4090 LET N$(K)=N$(A)
4100 LET I$(K)=I$(A)
4110 LET A$(K)=A$(A)
4120 LET T$(K)=T$(A)
4130 LET F$(K)=F$(A)
4140 LET C$(A)=""
4150 LET N$(A)=""
4160 LET I$(A)=""
4170 LET A$(A)=""
4180 LET T$(A)=""
4190 LET F$(A)=""
4200 LET A=A-1
4210 CLS
4220 PRINT AT 5,4;"NOMINATIVO CA
NCELLATO"
4230 PAUSE 125
4240 POKE 16437,255
4250 RETURN
5000 PRINT AT 5,1;"VUOI VERAMENT
E AZZERARE ? <S/N>"
5010 INPUT Y$
5020 IF Y$<>"S" AND Y$<>"N" THEN
 GOTO 5010
5030 IF Y$="N" THEN RETURN
5040 CLS
5050 RUN 10
6000 PRINT AT 4,3;"REGISTRAZION
E DATI";AT 10,2;"PREMI <N/L>,
<M>=MENU"
6010 INPUT Y$
6020 IF Y$="M" THEN RETURN
6030 CLS
6040 PRINT AT 5,0;"RIAVVOLGI IL
NASTRO";AT 8,0;"QUANDO SEI PRONT
O PER REGISTRAREPREMI <N/L>"
6050 INPUT Y$
6060 SAVE "RUBRICA"
6070 CLS
6080 GOTO 130
```

SERVIZIO SOFTWARE

Bit propone ai propri lettori i dischi o le cassette dei programmi pubblicati. I programmi, provati e garantiti, sono di immediato utilizzo.

Bit n°	Programma	Sistema	Prezzo	Codice	Supporto
38	Gioco della scimmia Spaccamattoni	VIC 20	15.000	VI381A	Cassetta
38	Text-Editor	PET 3032	20.000	PE381B PE382B	Cassetta Disco
38	Planel	Apple II	20.000	AP382C	Disco
39	Rompicapo di Rubik	CBM 4032 CBM 3032	15.000 20.000	PE391A PE392B	Cassetta Disco
39	Breakout	CBM 3032	20.000	PE393A	Cassetta
40	Reporter	Apple II	20.000	AP402C	Disco
41	Discover	Apple II	20.000	AP412C	Disco
42	Apple-Chef	Apple II	20.000	AP422C	Disco
42	Provariflessi	VIC 20	15.000	VI421A	Cassetta
45	Tiny FORTH	Apple II	35.000	AP452A	Disco
45	Alì Babà	ZX Spectrum	15.000	SP451B	Cassetta
45	1X2	PET 3032	15.000	PE451C	Cassetta

Per richiedere i programmi in contrassegno, pagando direttamente al postino la cifra indicata, inviare il seguente tagliando

Spedire in busta chiusa a Gruppo Editoriale Jackson - Via Rosellini, 12 - 20124 Milano

Inviatemi i seguenti nastri e/o dischi con i programmi pubblicati su Bit.

GRUPPO EDITORIALE JACKSON

Cod. ☐☐☐☐☐☐ **a L.** ....

Cod. ☐☐☐☐☐☐ **a L.** ....

Cod. ☐☐☐☐☐☐ **a L.** ....

Cod. ☐☐☐☐☐☐ **a L.** ....

Cognome ....

Nome ....

Indirizzo ....

CAP ....

Città ....

Spese postali (contributo fisso) L. 2.000

TOTALE L. ....

che pagherò al postino alla consegna del pacco.

Firma ....